ESP Networking

5/23/22 Brent Roman brent@mbari.org

SSH: Secure SHell

- Usual method of connecting
- Encrypted and secure
- May authenticate with password or key file
  - skip passwords by installing a key on your ESP
    - allowing your laptop unfettered access to it
- Data transfer with SSH
  - scp command copies files and whole directories
  - sftp supports Mozilla's "Filezilla" package
    - for point-and-click file editing & transfers

Telnet

- Very old, insecure, simple alternative to ssh
- Every computer OS has a telnet client
- Cannot transfer data files
- All text, including passwords sent unencrypted
- Should be disabled if ESP is directly on the internet
  - enabled by default
  - to disable, comment out telnet line in config file:

    `/etc/inetd.conf`
  - may be disabled by default in future

http: Hypertext Transfer Protocol

- Every ESP runs a local web server
  - Surf to:

    `http://`*`ESPhostname`*
  - Allows browsing of all files under `/var/log`
  - Does not allow uploading files
- Click on directory links to descend into them
- Does not work over VPNs
  - Upsync/upload logs to ESPshore, then Surf to:

    `http://ESPshore.mbari.org/`*`ESPname`*
    - example:
      - `http://ESPshore.mbari.org/waldo`

ftp: File Transfer Protocol

- Very old internet protocol
  - insecure, difficult to route over tunnels
  - ESP does not allow uploading files via FTP
  - anonymous (user ‘ftp’) accesses files under

    `/var/log` #same as http
  - other users can log in to download their files
- Modern web browsers dropped FTP support
- User login should be disabled if ESP on the internet
  - comment out ‘local_enable’ line in config file:

    `/etc/vsftpd.conf`
  - this may be disabled by default in future

ESPshore.mbari.org

- Hub of ESP VPN (Virtual Private Network)
  - provides access to ESPs:
    - on public networks
    - on networks behind restrictive firewalls
    - on networks lacking local DNS
- Data repository
  - ESPs periodically “upsync” data to shore
  - Access to data on ESPshore is **public** !
- Routes email from ESPs via Gmail account

  `mbariesp@gmail.com`

ESP VPN: Virtual Private Network

- Many networks provide internet access, but
  - don’t allow any incoming connections
- ESPshore accepts incoming ESP connections
  - and tunnels all traffic inside that physical connection
  - thus creating a Virtual Private Network
    - private, because all the virtual network traffic is encrypted
- VPN assigns each ESP a unique virtual IP address
- ESPs may access each other by virtual IP or name
  - Access between ESPs owned by different groups
    - normally blocked by ESPshore
    - may be allowed, if groups agree

Alternative ESP VPN Technologies

- Wireguard tunnels traffic over UDP
  - (User Datagram Packets)
  - very fast, secure, and robust
  - Not supported by older ESPs
  - Blocked only by overly restrictive firewalls
- PPtP: Point-of-Presence tunneling Protocol
  - tunnels traffic over PPP
  - Obsolete – no longer permitted at MBARI
  - Encrypted, but not as secure as others
  - Blocked by many corporate firewalls

SSH tunneling

- VPN like behavior
  - for situations where Wireguard is blocked
  - may be layered over another VPN
  - tunnels only certain protocols over SSH
    - very secure
    - slow and not robust
    - Not a true Virtual Network
      - Does not provide Virtual IP addresses
    - Does not handle older protocols like FTP
    - Each new protocol requires additional work to support

Outside Access to VPN'd ESPs

- `ESPshore.mbari.org` is the hub for all outside access
  - But, ESPshore has only one IP address
    - to apportion among dozens of ESPs
    - Individual ESPs assigned unique TCP port numbers
      - typically within the range of 2000 – 2998
    - Not the "well known" port numbers normally used
      - SSH, HTTP, FTP protocols default to "well known ports"
      - Must override those default port numbers to select particular ESPs
  - URLs may be followed by : to indicate a TCP port #
    - Try: http://www.google.com:80 #80 = http well known port#

      Try: http://www.google.com:2000 #does not connect
  - Some networking programs allow overriding TCP ports
    - ssh and telnet, for instance

ESP Access Example

- This example assumes use of OpenSSH
  - default SSH for Unix, Linux, and MacOS
  - available on newer Windows releases
- ESPbrent is a ESP CPU board in my home
  - SSH access is via port 2008 on ESPshore
  - Try:

```
$ ssh -p 2008 esp@ESPshore.mbari.org
```

- Did you get this warning?

```
WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!
```

SSH Remote Host Identification

- SSH remembers Host Identification keys
  - when reconnecting to same Host/IP addr
    - SSH expects the key to match previous
  - It doesn’t when many Hosts are accessed
    - via different TCP ports on a single IP
      - e.g. `ESPshore.mbari.org`
  - SSH thinks ESPshore is impersonating all the ESPs behind it.
    - In a way, it is :-)

Suppress Host ID warnings

- Tell OpenSSH to not to remember Host IDs
  - Try:

```
$ ssh -qp 2008 -o UserKnownHostsFile=/dev/null esp@ESPshore.mbari.org
```

  - Works, but:
    - inconvenient to type
    - subverts all Host ID checking
  - There must be a better way...

Configuring OpenSSH

- Create an ssh config file
  - defining aliases for all your ESPs
  - so you can ssh to them conveniently
- ssh config file has (too) many options:

  `https://linux.die.net/man/5/ssh_config`
- create a `config` text file inside

  `$HOME/.ssh`

Bad Example ~/.ssh/config

```
#Bad example of OpenSSH config to access ESPshore
Host ESPshore espshore
HostName ESPshore.mbari.org
UserKnownHostsFile=/dev/null
ForwardX11  no
```

- Test it:

```
ssh -p 2008 esp@ESPshore
```

- Still requires you to remember the magic port numbers
- Still disables security against impersonation
- We can do better!

Better Example ~/.ssh/config

```
#Example of OpenSSH config to access ESPbrent
Host brent.vpn
HostName ESPshore.mbari.org
Port 2008
CheckHostIP no
ForwardX11  no
```

- Test it:

```
ssh esp@brent.vpn
```

- No warnings
- No magic port numbers to remember
- Preserves security against impersonation

ESP Email

- Automatic emails generated on:
  - BadNews – when ESP stops due to an error
  - GoodNews – general ESP progress email
  - Notice – new phase begun
  - HABfans|BACfans|*etc* – email about specific protocols
- Email configured in `mission/phasecfg.rb` file
- Initially all emails sent to all ESP (zoo:-) wardens:
  - However, individuals can be removed from specific lists

    `-> BadNews.remove “BigBossMan@megacorp.com”`
  - People can be later added to all email lists with:

    `-> Email.add “NewGuy@dues.org”`
- Emails sent only when `Thread.realtime? == true`

ESP Email Configuration in mission/phasecfg.rb

```
Email.server = 'mail'
Email.domain = 'mbari.org'  #default email domain if none specified
#comment next line out if operating on MBARI's network
Email.account = {:user=>"mbariesp", :password=>"esps@MBARI", :auth=>:login}

#comment next line out to disable pushing logs to ESPserver after email sent
Log.uploadCmd = "upsync"  #shell command used to upload logs

mail = Thread.realtime? ? Email : Email::Suppressed
#mail = Email::Suppressed

wardens = ['brent']
#wardens |= ['esp001','preston','kyamahara']

:BadNews.denotes mail.new wardens, :Subject=>'Trouble'
:GoodNews.denotes mail.new wardens.dup, :Subject=>'Progress'
:Notice.denotes mail.new wardens.dup, :Subject=>'Notice'
```

Managing Email during mission

- *EmailList* = BadNews|GoodNews|Notice|*fans
  - where * is a protocol name (e.g. HABfans)
- Add/Remove addresses from *EmailList*:

  -> *EmailList*.add “list”,”of”,”email”,”addresses”

  -> *EmailList.remove* “list”,”of”,”email”,”addresses”
- Add/Remove addresses from all Email lists

  -> Email.add “joe@corp.com”,”jane@green.org”

  -> Email.*remove* “list”,”of”,”email”,”addresses”
- List all *EmailLists*

  -> Email.all
- List addresses on given *EmailList*

  -> *EmailList*.add
- Send short message to given *EmailList*

  -> *EmailList*.email “message text”

upsync: Uploading to ESPshore

- Uses rsync to efficiently upload only changed data
  - from an ESP to ESPshore when an email is sent
  - rsync is Unix utility for Remote file Sychronization
- upsync is a shell script defined in esp2/bin
  - invoked within the ESP app as:

```
-> Log.upload.finish #await upload completion
```

    - uploadCmd must have been enabled in phasecfg.rb
  - customization is possible
    - which files and directories to upload
    - whether to use ssh or to rely on being inside VPN
    - how to detect file changes
      - Can we assume files are always appended to?
      - If so, we can upload much less data

Invoking upsync from Linux Shell

```
$ upsync --help
Upload ESP logs to ESPbrent@ESPshore:/home/ftp/brent/esp
First args starting with - are additional rsync options
Remaining args are files and directories in /var/log/esp overriding default list
Examples:  (11/10/21 brent@mbari.org)
  upsync  #uploads simfast simfast.* *.spr* *.pcr* *.tif* CTD-*.hex .././messages
  upsync -v     #upload above files verbosely
  upsync simfast.log    #upload just the binary log file
  upsync '*.spr'    #upload all spr files
Environment variables:
 $shore    = rsync destination (ESPbrent@ESPshore:/home/ftp/brent/esp)
 $logHome  = source directory  (/var/log/esp)
 $logFiles = default file list (...)
 $rsync    = rsync cmd+options (rsync --inplace -rzRe "ssh -yp2323")
```

Configuring upsync

Carefully edit the script esp2/bin/upsync:

```
#uncomment next line for minimal upsync (via Iridium)
#: ${logFiles:=$mode.phase $mode.puck $mode.out CTD-*.hex .././messages}
: ${logFiles:=$mode $mode.* *.spr* *.pcr* *.tif* CTD-*.hex .././messages}
: ${logHome:=/var/log/$USER}
#uncomment next line for upsync in ssh tunnel to rsync daemon on shore
#: ${rsync:=rsync --append -rzR}  ${shore:=shore::$ESPname/$USER}
: ${rsync:=rsync --inplace -rzRe \"ssh -yp2323\"}
: ${shore:=`hostname`@ESPshore:/home/ftp/$ESPname/$USER}
```

Add midres ESP2 image directory to files uploaded by default:

```
: ${logFiles:=$mode $mode.* *.spr* *.pcr* *.tif* CTD-*.hex .././messages midres}
```

Dramatically increase upload speed by assuming all files are append only:

```
: ${rsync:=rsync --append -rzR}  ${shore:=shore::$ESPname/$USER}
```

Browsing on ESPshore

Point any web browser at:

`http://ESPshore.mbari.org/`*`ESPname`*`/esp`

For example:

`http://ESPshore.mbari.org/friday/esp`

Configure your browser to process file associations usefully:

.tiff => imageJ

SFTP: Secure shell FTP

- Uses SSH to securely transfer files similarly to FTP
  - Many file manipulation GUIs support SFTP
  - May both read and overwrite files (be careful)
- Example: Configuring Filezilla to access esp@brent.vpn
  - Create new site in “Site Manager” dialog
    - name it “esp@brent.vpn”
    - Under ‘General’ Tab
      - Protocol: SFTP
      - Host: ESPshore.mbari.org
      - Port: 2008
      - Logon Type: Ask for password
      - User: esp
  - Click “Connect” button near bottom of dialog

FTP to ESPshore with Filezilla

- Browse uploaded data without fear of overwriting
- Configuring Filezilla to access ESPshore via anonymous FTP
  - Create new site in “Site Manager” dialog
    - name it “ESPshore”
    - Under ‘General’ Tab
      - Protocol: FTP
      - Host: ESPshore.mbari.org
      - Port: *<LEAVE BLANK>*
      - Encryption: Only use plain FTP (insecure)
      - Logon Type: Normal
      - User: ftp
  - Click “Connect” button near bottom of dialog