Using Carbon – Part II

Remote access

Michael Sternberg

Access Policies

- Login nodes
  - from inside Argonne (incl. VPN): directly

    `clogin.cnm.anl.gov`
  - from offsite: through ssh gateway

    `mega.cnm.anl.gov`
  - Access using Argonne domain account
  - DOE-compliant password/passphrase* required

- Compute nodes
  - normally via PBS jobs only
  - interactive ssh possible – when *your job* is running

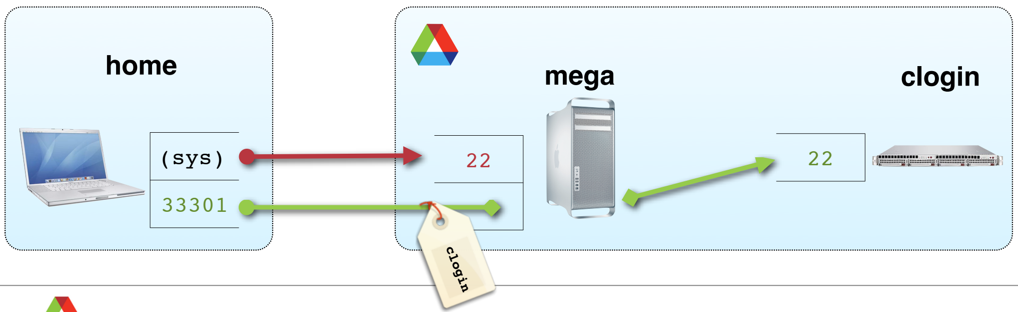
SSH gateway access – Getting started

- Establish a test tunnel, register gateway

```
home window1> ssh -L 33301:clogin:22 \
                   argonne_id@mega.cnm.anl.gov
```

- Connect through the tunnel, register target host

```
home window2> ssh -p 33301 argonne_id@localhost
```



Tunnel configuration – OpenSSH

- On home machine, create or add to `~/.ssh/config`

```
NoHostAuthenticationForLocalhost yes

Host mega
    Hostname             mega.cnm.anl.gov
    User                 argonne_id
    LocalForward         33301     clogin:22
    LocalForward         33343     wiki.inside.anl.gov:443
    LocalForward         33380     cmgmt1:80

Host carbon
    Hostname             localhost
    User                 argonne_id
    Port                 33301
    ForwardX11           yes
    ForwardX11Trusted    yes
```

host aliases

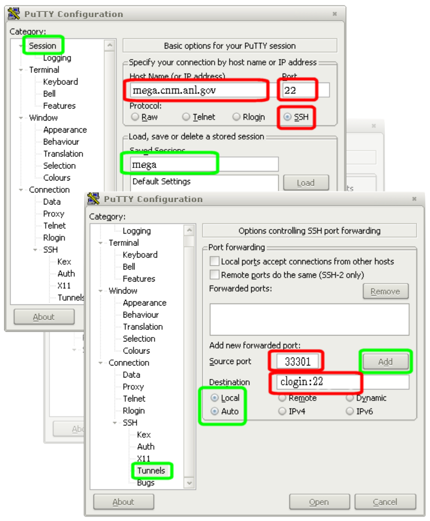
Bonus: intranet web access

https://localhost:33343/
http://localhost:33380/

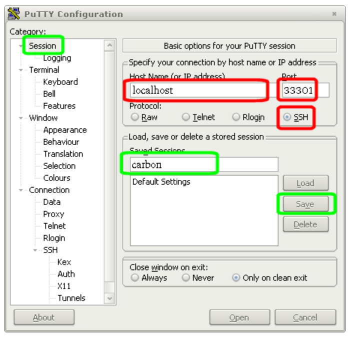
- sample in `/home/share/network/ssh-config.sample`

Tunnel configuration – Putty

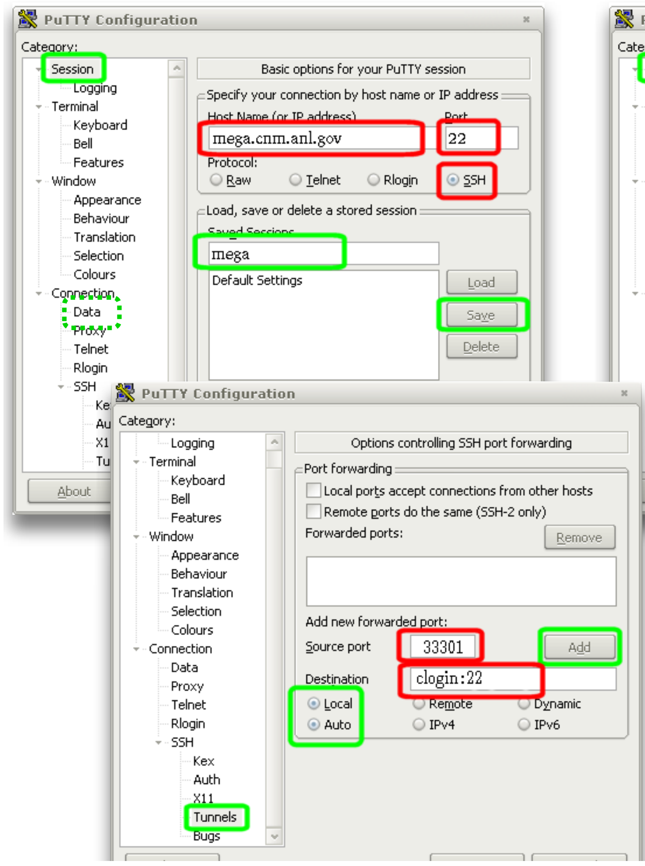
- configure *sessions*


PuTTY Configuration — Session
Basic options for your PuTTY session
Specify your connection by host name or IP address
Host Name (or IP address): mega.cnm.anl.gov
Port: 22
Protocol: Raw Telnet Rlogin [✓ SSH]
Load, save or delete a stored session
Saved Sessions: mega

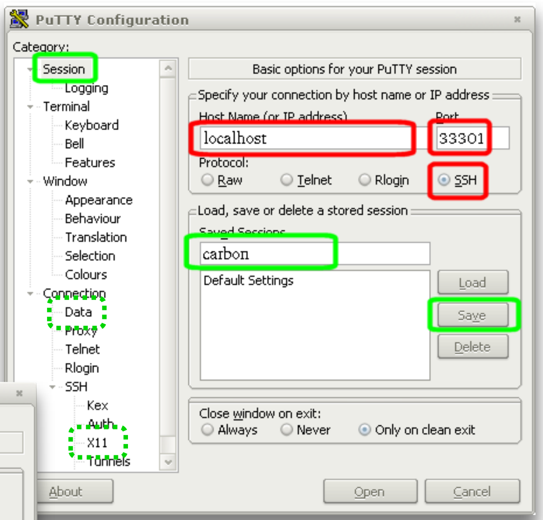
PuTTY Configuration — Connection > SSH > Tunnels
Options controlling SSH port forwarding
Port forwarding
Add new forwarded port:
Source port: 33301
Destination: clogin:22
[✓ Local] Remote Dynamic
[✓ Auto] IPv4 IPv6
Add


PuTTY Configuration — Session
Host Name (or IP address): localhost
Port: 33301
Protocol: Raw Telnet Rlogin [✓ SSH]
Saved Sessions: carbon
Save

putty tunnel
Web Results 1 - 10 of about 138,000


PuTTY Configuration — Session
Host Name (or IP address): mega.cnm.anl.gov
Port: 22
Protocol: Raw Telnet Rlogin [✓ SSH]
Saved Sessions: mega
Save

PuTTY Configuration — Connection > SSH > Tunnels
Add new forwarded port:
Source port: 33301
Destination: clogin:22
[✓ Local] Remote Dynamic
[✓ Auto] IPv4 IPv6
Add


PuTTY Configuration — Session
Host Name (or IP address): localhost
Port: 33301
Protocol: Raw Telnet Rlogin [✓ SSH]
Saved Sessions: carbon
Save
Close window on exit: Always Never [✓ Only on clean exit]

http://www.unixwiz.net/
techtips/putty-openssh.html

Tunnel use

- (Re-)establish tunnel

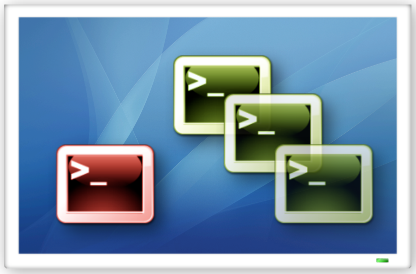
```
home window1> ssh -v mega
```

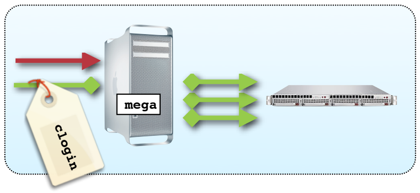
- Open login session

```
home window2> ssh carbon
```

- Multiple uses, plus file transfer

```
home window3> ssh carbon
home window4> scp -p file  carbon:path/to/dir/
home window4> scp -p carbon:path/to/dir/file  .
```





Other tunnel applications

- Interactive file transfer

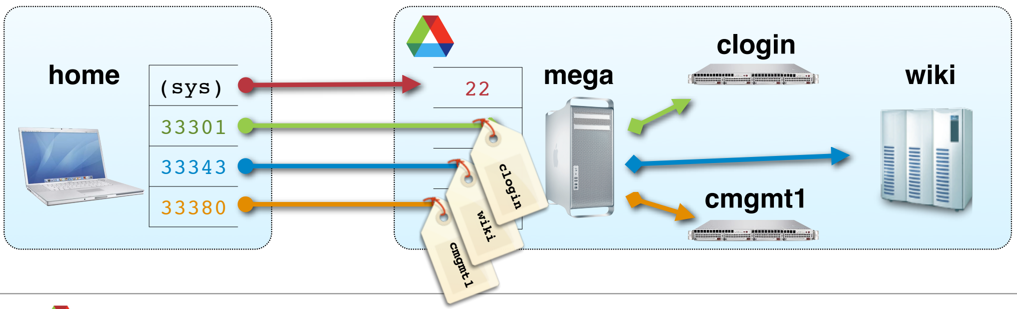
```
home> sftp carbon
```

- Cluster status

```
http://localhost:33380/ganglia
```

- Web documentation

```
https://localhost:33343/cnm/HPC
```



ssh public keys – concept

- Encryption of data traffic
  - Locked by one key, unlocked by another
  - Key stored as separately encrypted file
  - *Agent* enables password-less connections
- Initialization
  1. Create ssh key pair
  2. Copy public key to destination system(s)
  3. Type passphrase into agent – *once per desktop session*
- Same principle across all ssh implementations
  - details vary – mostly in *ssh-agent* startup

ssh public keys – initialization

- Create ssh key pair
  - *use a strong passphrase*

```
home> ssh-keygen -t rsa
...
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/home_id/.ssh/id_rsa.
Your public key has been saved in /Users/home_id/.ssh/id_rsa.pub.
...
```

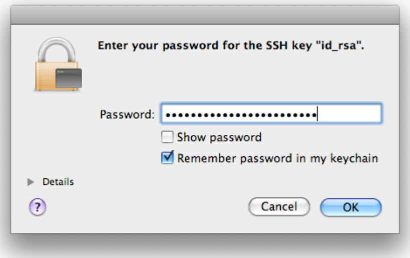
- Copy *public key* to Carbon*

```
home> scp -p ~/.ssh/id_rsa.pub   carbon:

login1> cat ~/id_rsa.pub >> ~/.ssh/authorized_keys
login1> rm ~/id_rsa.pub
```

ssh-agent – general operation

- Usually started by OS on login – when keys are present
- Negotiates access for each connection
  - *Mac OS X 10.4 – Tiger*
    - install SSHKeychain.app, from www.sshkeychain.org
    - no longer updated, still usable
  - *Mac OS X 10.5 – Leopard*
    - automatic, using Keychain.app*


Enter your password for the SSH key "id_rsa".

Password: ••••••••••••••••••••••••

[ ] Show password

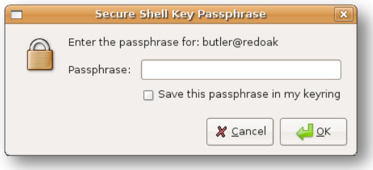
[✓] Remember password in my keychain

Demo

ssh-agent – Linux, Windows

- *Linux (Gnome, KDE, ...)*
  - usually automatic

    (gnome-ssh-askpass or similar)

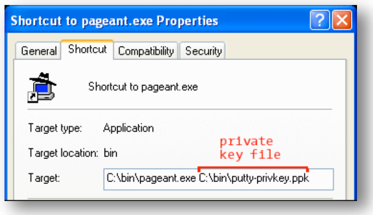

Secure Shell Key Passphrase

Enter the passphrase for: butler@redoak

Passphrase:

[ ] Save this passphrase in my keyring

- *Windows*
  - use *Pageant* from the *Putty* suite
    - http://www.chiark.greenend.org.uk/~sgtatham/putty/
  - instructions:
    - http://www.unixwiz.net/techtips/putty-openssh.html


Shortcut to pageant.exe Properties

Shortcut to pageant.exe

Target type: Application

Target location: bin

Target: C:\bin\pageant.exe C:\bin\putty-privkey.ppk

Remote file access

- File transfer methods
  - Command line – *for bulk and batch*
    - scp, sftp, rsync
  - GUI applications – *quirky*
    - Fugu, Cyberduck, WinSCP
- File system mounts
  - Linux FUSE, MacFUSE, SftpDrive

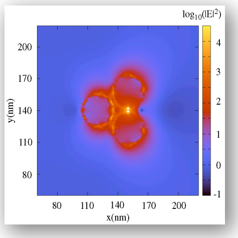
```
sshfs  carbon:  /tmp/carbonhome
```

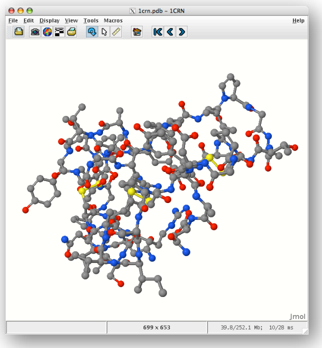
  - *full semantics; stat(2) is slow*
    - cp, mv, rm, … also: ln, open in applications

Demo

Analysis and visualization – on-cluster vs. "at home" processing

- Analysis on cluster
  - access cluster with X11 forwarding
  - *run* graphics on cluster, *display* at home
- Analysis at home
  - transfer files or mount file system
  - run and display at home
- Factors
  - sizes: data vs. graphics; refresh rate
  - network bandwidth
  - software: availability, usability
  - turnaround time





Remote graphics using X11

- *X11 server* must be running on *home* machine
- X11Forwarding – already configured

```
home:~/.ssh/config
```

- Verify setup and functionality:

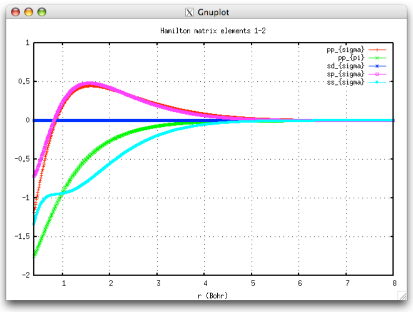
```
login1> echo $DISPLAY
localhost:14.0
```

  - (details vary by connection)

```
login1> xload
```

- Start applications:

```
gnuplot, rasmol, vmd,
...
```



Gnuplot frontend "pl"

- Quickly fire up gnuplot for routine analysis

```
login1> pl --help
...
Usage: pl [-tty|-ps|-eps|-cps|-term term] [-set option] [-u using] [-w style]
       [...]  [datafile ...]

    Reads data from stdin or specified file[s] and feeds them to the
    gnuplot "plot" command, either simultaneously or in sequence [-seq].

    Data file interpretation:
    - may contain "y" only, "x y", "x y1 y2 .."
    - comments ("#" first on line) and empty lines permitted.
...
    - embedded lines starting with "#@" are used as gnuplot commands
      just prior to plotting current file
    - optional embedded column labels as "# columns: label1 | label2 ..."

    Gnuplot commands are merged (in this order) from:
    internal defaults, ~/.plrc, .plrc, command line, data files
...
Author:  Michael Sternberg <sternberg@phys.uni-paderborn.de>, 1995-2003
```

Demo