Tutorial: SSH

Secure SHell: Connect remotely anything, anywhere

UL High Performance Computing (HPC) Team

Sebastien Varrette

University of Luxembourg (UL), Luxembourg
http://hpc.uni.lu

UNIVERSITÉ DU LUXEMBOURG

Summary

Summary

SSH: Secure Shell

- Ensure **secure** connection to remote (UL) server
  - ↪ establish **encrypted** tunnel using **asymmetric keys**
    - ✓ **Public** `id_rsa.pub` **vs. Private** `id_rsa` (**without** `.pub`)
    - ✓ typically on a non-standard port (**Ex**: 8022) *limits kiddie script*
    - ✓ Basic rule: 1 machine = 1 key pair
  - ↪ the private key is **SECRET**: **never** send it to anybody
    - ✓ Can be protected with a passphrase

SSH: Secure Shell

- Ensure **secure** connection to remote (UL) server
  - ↪ establish **encrypted** tunnel using **asymmetric keys**
    - ✓ **Public** `id_rsa.pub` **vs. Private** `id_rsa` (**without** `.pub`)
    - ✓ typically on a non-standard port (**Ex**: 8022) *limits kiddie script*
    - ✓ Basic rule: 1 machine = 1 key pair
  - ↪ the private key is **SECRET**: **never** send it to anybody
    - ✓ Can be protected with a passphrase

- SSH is used as a secure backbone channel for **many** tools
  - ↪ Remote shell **i.e** remote command line
  - ↪ File transfer: `rsync`, `scp`, `sftp`
  - ↪ versionning synchronization (`svn`, `git`), github, gitlab etc.

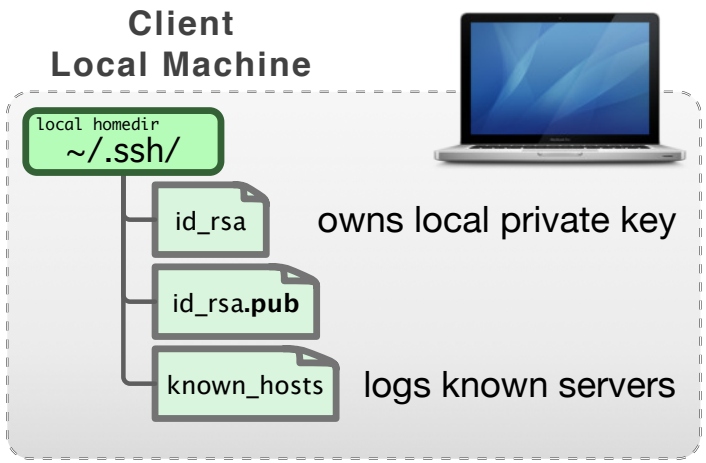
SSH: Secure Shell

- Ensure **secure** connection to remote (UL) server
  - ↪ establish **encrypted** tunnel using **asymmetric keys**
    - ✓ **Public** `id_rsa.pub` **vs. Private** `id_rsa` (**without** `.pub`)
    - ✓ typically on a non-standard port (**Ex**: 8022) *limits kiddie script*
    - ✓ Basic rule: 1 machine = 1 key pair
  - ↪ the private key is **SECRET**: **never** send it to anybody
    - ✓ Can be protected with a passphrase

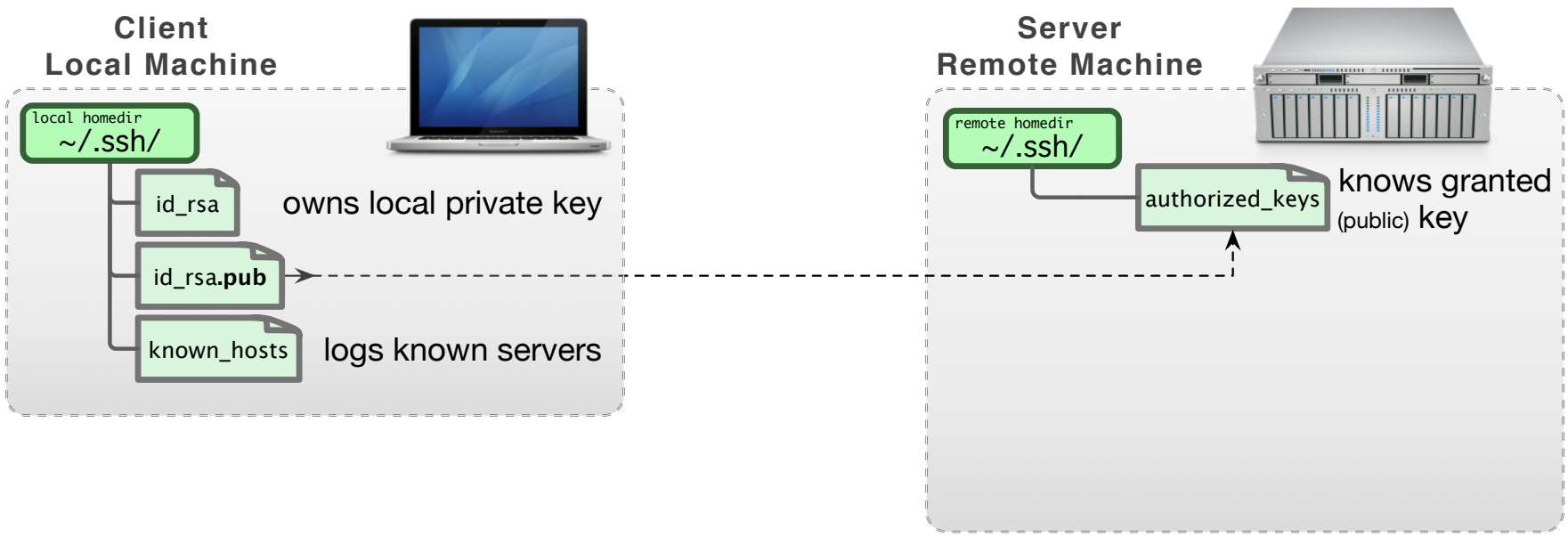
- SSH is used as a secure backbone channel for **many** tools
  - ↪ Remote shell **i.e** remote command line
  - ↪ File transfer: `rsync`, `scp`, `sftp`
  - ↪ versionning synchronization (`svn`, `git`), github, gitlab etc.

- Authentication:
  - ↪ ~~password~~ (disable if possible)
  - ↪ (*better*) **public key authentication**

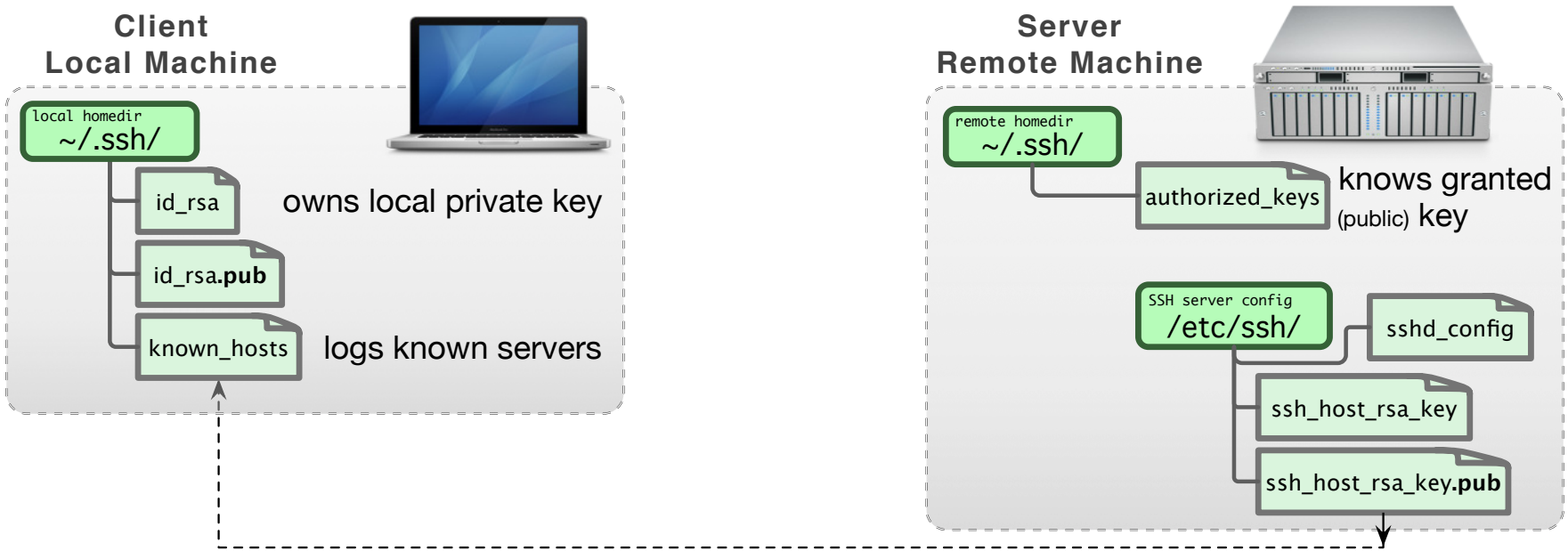
SSH: Public Key Authentication



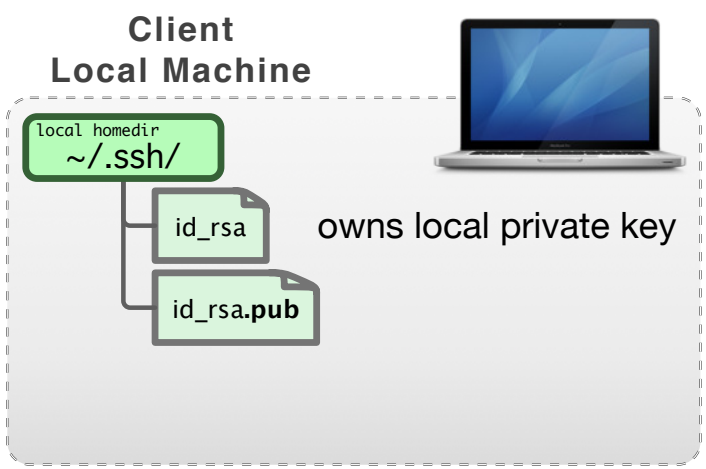
SSH: Public Key Authentication

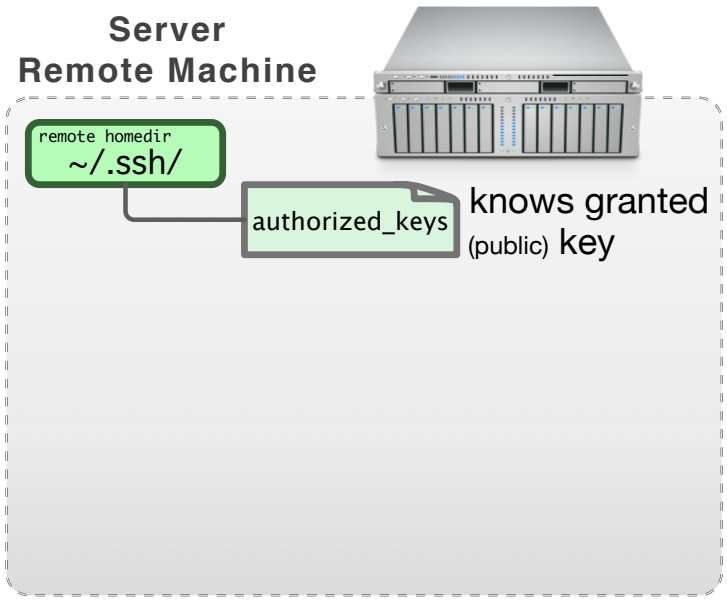


SSH: Public Key Authentication

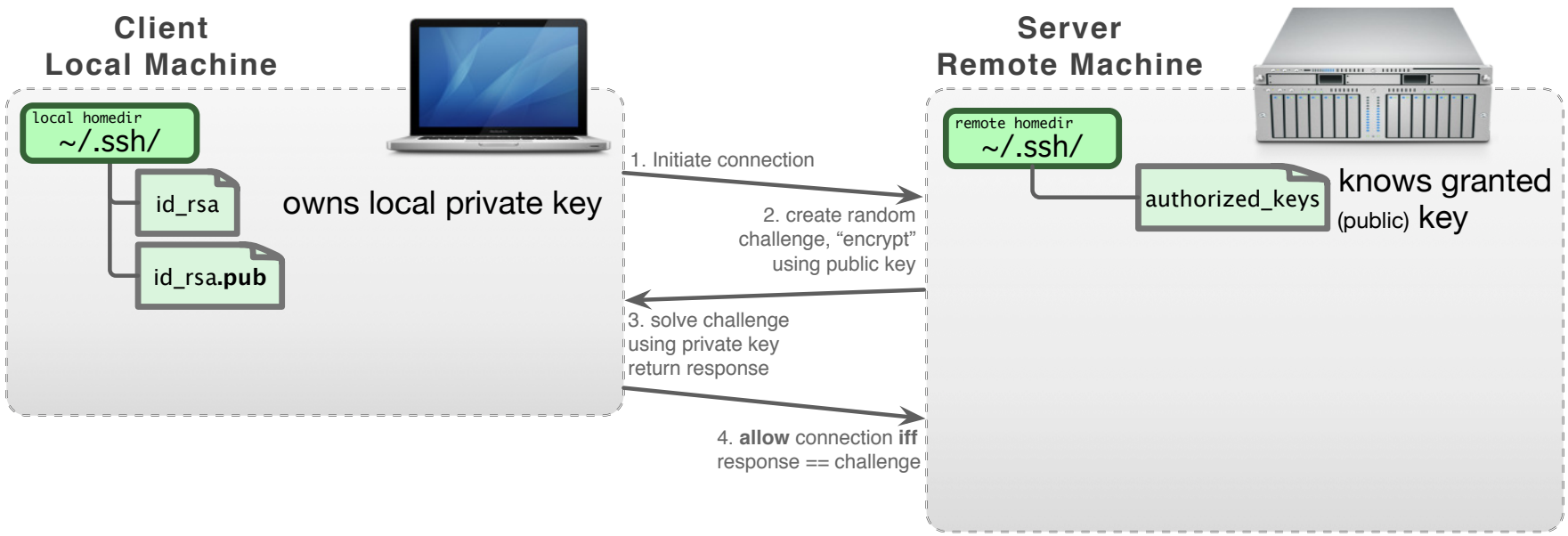


SSH: Public Key Authentication





SSH: Public Key Authentication



- **Restrict** to public key authentication: `/etc/ssh/sshd_config`:

```
PermitRootLogin no
# Disable Passwords
PasswordAuthentication no
ChallengeResponseAuthentication no
```

```
# Enable Public key auth.
RSAAuthentication yes
PubkeyAuthentication yes
```

Summary

SSH Setup on Linux / Mac OS

- OpenSSH natively supported; configuration directory : `~/.ssh/`
  - ↪ package `openssh-client` (Debian-like) or `ssh` (Redhat-like)
- SSH Key Pairs (public vs private) generation: `ssh-keygen`
  - ↪ specify a **strong** passphrase
    - ✓ protect your **private** key from being stolen **i.e.** impersonation
    - ✓ **drawback:** passphrase must be typed to use your key

SSH Setup on Linux / Mac OS

- OpenSSH natively supported; configuration directory : `~/.ssh/`
  - ↪ package `openssh-client` (Debian-like) or `ssh` (Redhat-like)
- SSH Key Pairs (public vs private) generation: `ssh-keygen`
  - ↪ specify a **strong** passphrase
    - ✓ protect your **private** key from being stolen **i.e.** impersonation
    - ✓ ~~**drawback:** passphrase must be typed to use your key~~ `ssh-agent`

SSH Setup on Linux / Mac OS

- OpenSSH natively supported; configuration directory : `~/.ssh/`
  - ↪ package `openssh-client` (Debian-like) or `ssh` (Redhat-like)
- SSH Key Pairs (public vs private) generation: `ssh-keygen`
  - ↪ specify a **strong** passphrase
    - ✓ protect your **private** key from being stolen **i.e.** impersonation
    - ✓ ~~**drawback:** passphrase must be typed to use your key~~ `ssh-agent`

DSA and RSA 1024 bit are deprecated now!

SSH Setup on Linux / Mac OS

- OpenSSH natively supported; configuration directory : `~/.ssh/`
  - ↪ package `openssh-client` (Debian-like) or `ssh` (Redhat-like)
- SSH Key Pairs (public vs private) generation: `ssh-keygen`
  - ↪ specify a **strong** passphrase
    - ✓ protect your **private** key from being stolen **i.e.** impersonation
    - ✓ ~~**drawback:** passphrase must be typed to use your key~~ `ssh-agent`

DSA and RSA 1024 bit are deprecated now!

```
$> ssh-keygen -t rsa -b 4096 -o -a 100          # 4096 bits RSA
(better)  $> ssh-keygen -t ed25519 -o -a 100    # new sexy Ed25519
```

Private (identity) key

`~/.ssh/id_{rsa,ed25519}`

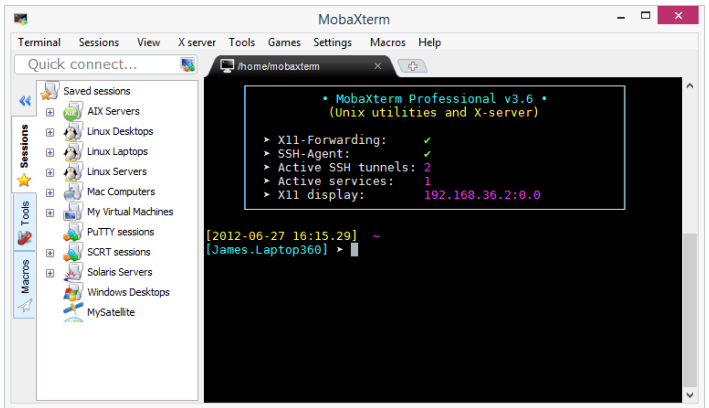
Public Key

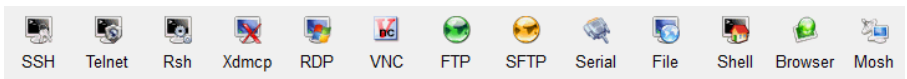
`~/.ssh/id_{rsa,ed25519}.pub`

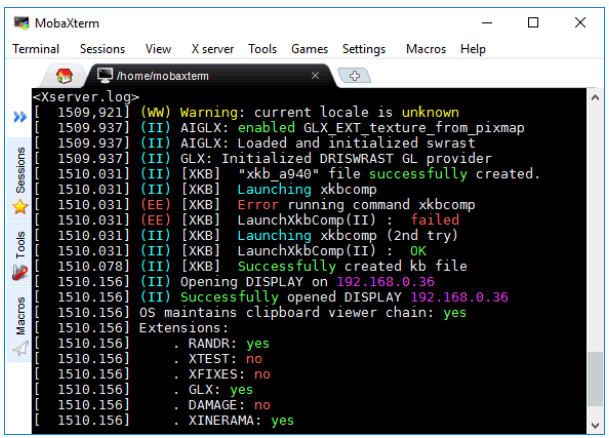
SSH Setup on Windows

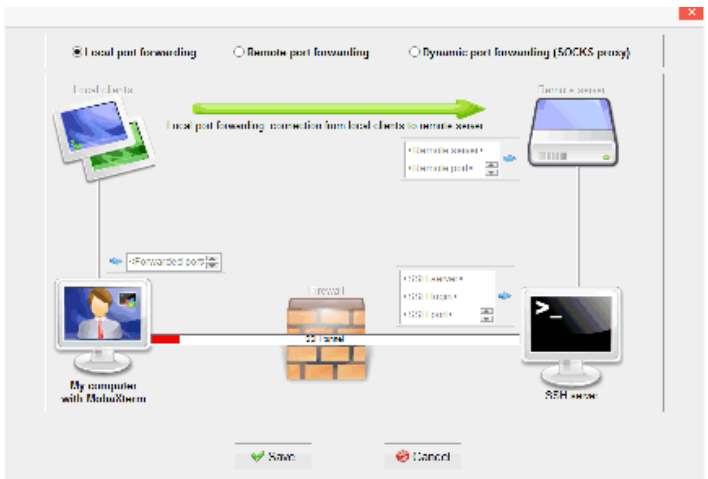
- Use MobaXterm!
  - ↪ [tabbed] Sessions management
  - ↪ X11 server w. enhanced X extensions
  - ↪ Graphical SFTP browser
  - ↪ SSH gateway / tunnels wizards
  - ↪ [remote] Text Editor
  - ↪ ...

http://mobaxterm.mobatek.net/



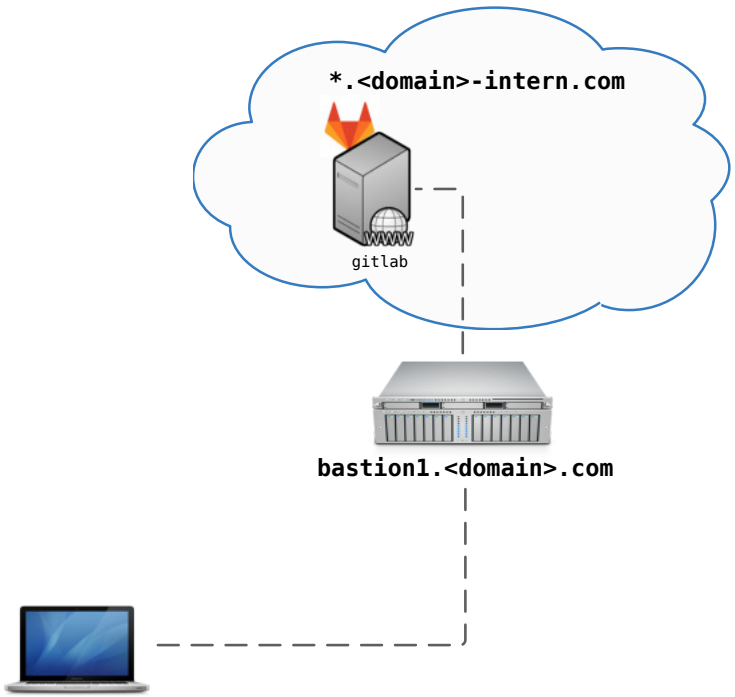




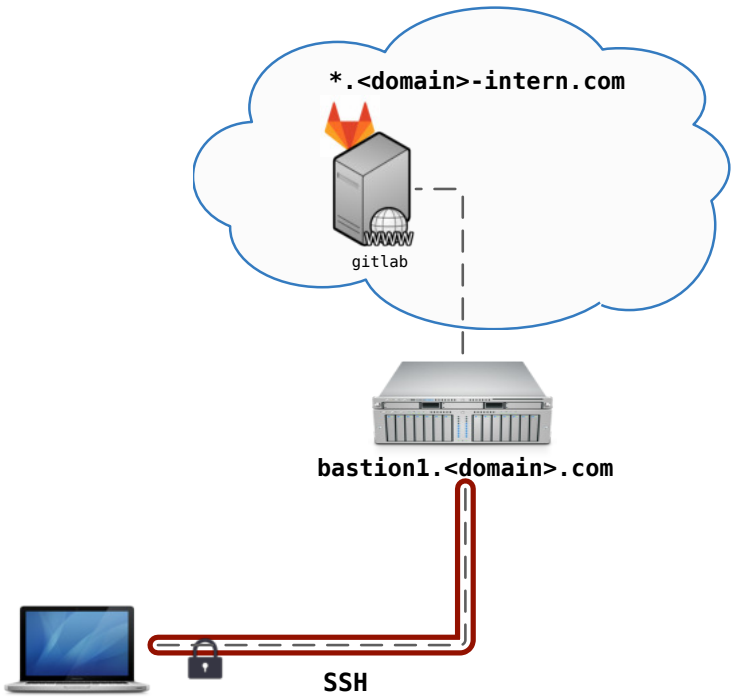


Summary

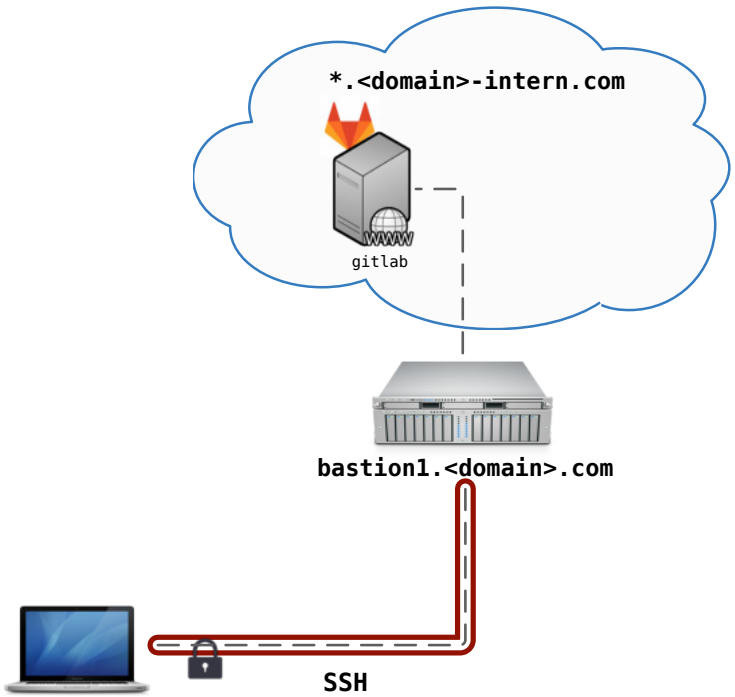
SSH Basic Usage



SSH Basic Usage



SSH Basic Usage

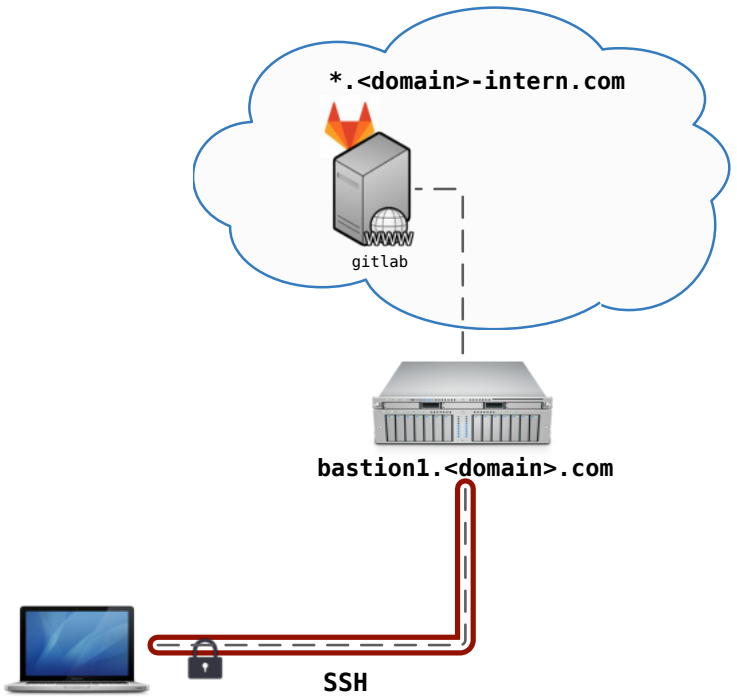


OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com
```

SSH Basic Usage



PuTTY / PLink / Pageant (Windows)

```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>
```

OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com
```

SSH in Practice ~/.ssh/config

```
$> ssh [-X] [-p <port>] <login>@<hostname>
# Example:  ssh -p 8022 svarrette@access-iris.uni.lu
```

```
Host <shortname>
    Port <port>
    User <login>
    Hostname <hostname>
```

- ~/.ssh/config:
  - ↪ Simpler commands
  - ↪ Bash completion

    $> ssh iri<TAB>

SSH in Practice ~/.ssh/config

```
$> ssh [-X] [-p <port>] <login>@<hostname>
# Example:  ssh -p 8022 svarrette@access-iris.uni.lu
```

```
Host *.ext_ul
    ProxyCommand ssh -q iris-cluster \
        -W `basename %h .ext_ul`:%p
# UL HPC Platform -- http://hpc.uni.lu
Host iris-cluster
    Hostname      access-iris.uni.lu
Host *-cluster
    User          login #ADAPT accordingly
    Port          8022
    ForwardAgent no
```

```
Host <shortname>
    Port <port>
    User <login>
    Hostname <hostname>
```

- ~/.ssh/config:
  - ↪ Simpler commands
  - ↪ Bash completion

    $> ssh iri<TAB>

SSH in Practice ~/.ssh/config

```
$> ssh [-X] [-p <port>] <login>@<hostname>
# Example:  ssh -p 8022 svarrette@access-iris.uni.lu
```

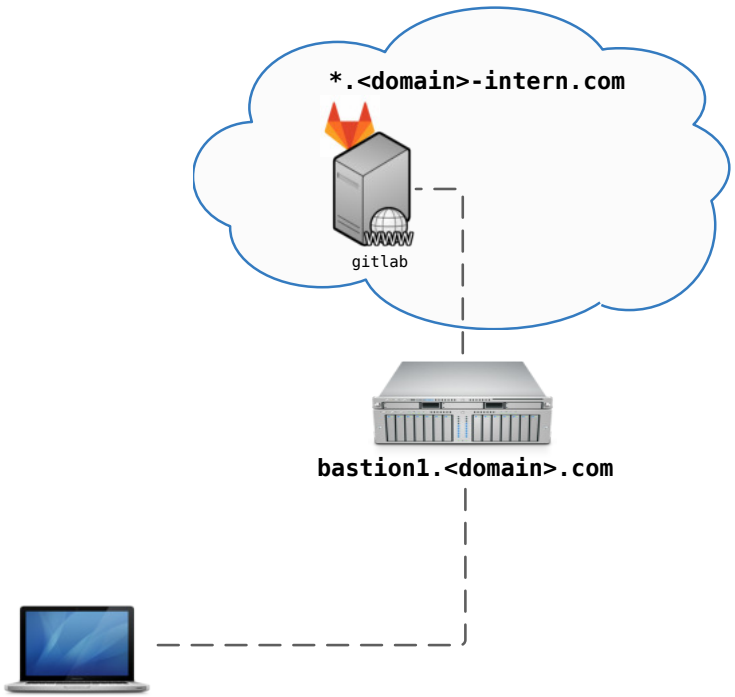
```
Host *.ext_ul
    ProxyCommand ssh -q iris-cluster \
        -W `basename %h .ext_ul`:%p
# UL HPC Platform -- http://hpc.uni.lu
Host iris-cluster
    Hostname      access-iris.uni.lu
Host *-cluster
    User          login #ADAPT accordingly
    Port          8022
    ForwardAgent no
```

```
Host <shortname>
    Port <port>
    User <login>
    Hostname <hostname>
```

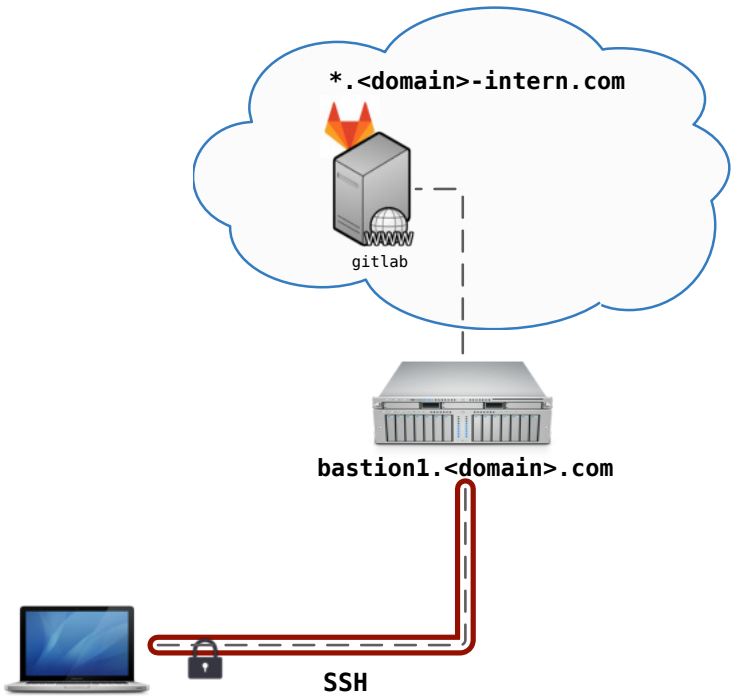
- ~/.ssh/config:
  - ↪ Simpler commands
  - ↪ Bash completion

```
$> ssh iri<TAB>
$> ssh iris-cluster
$> ssh work
$> ssh work.ext_ul
```

SSH Advanced Usage: SOCKS Proxy



SSH Advanced Usage: SOCKS Proxy



PuTTY / PLink / Pageant (Windows)

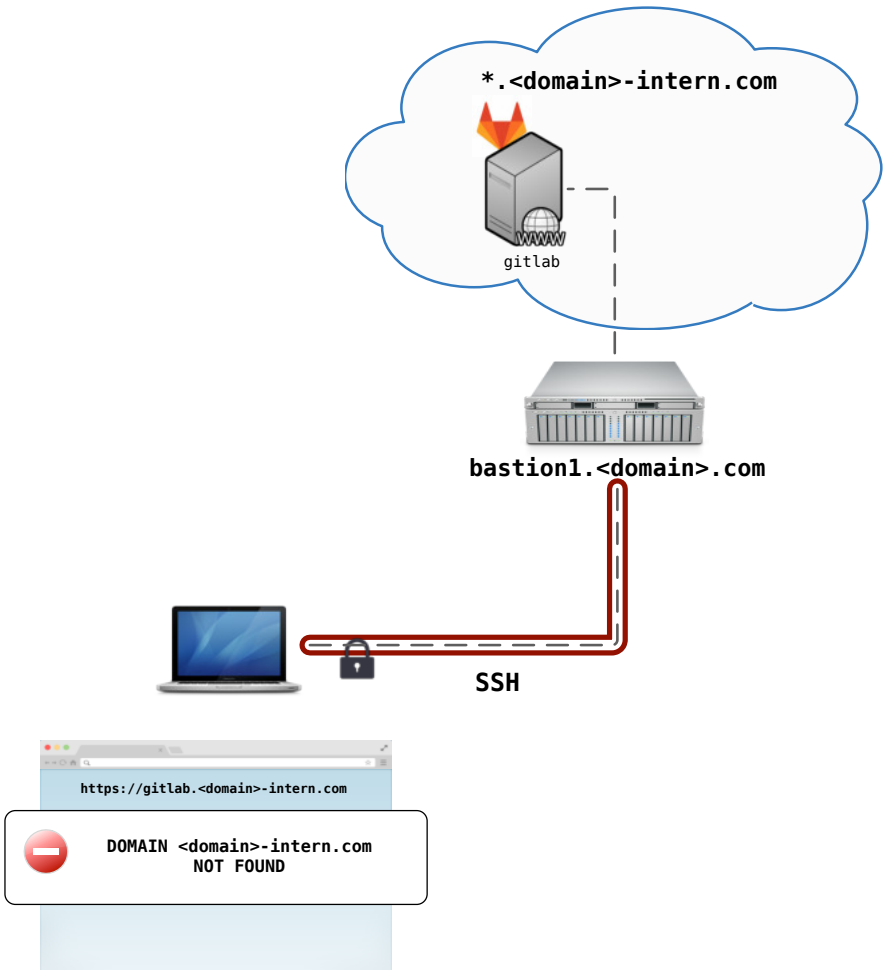
```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>
```

OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com
```

SSH Advanced Usage: SOCKS Proxy



PuTTY / PLink / Pageant (Windows)

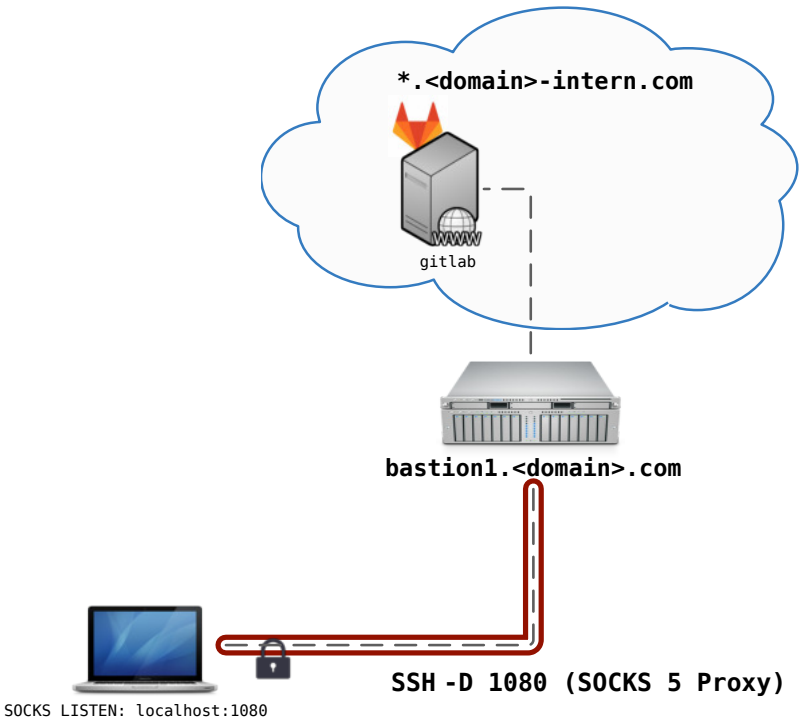
```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>
```

OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com
```

SSH Advanced Usage: SOCKS Proxy



PuTTY / PLink / Pageant (Windows)

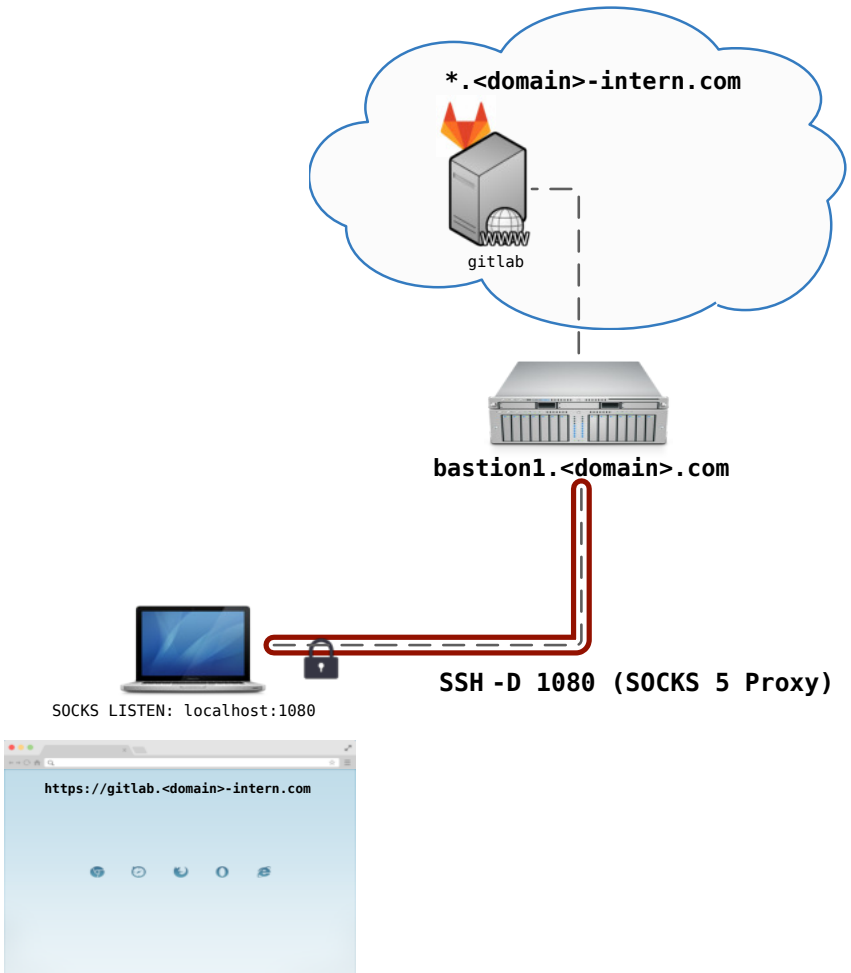
```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>
- Connection/SSH/Tunnels: Port 1080, Dynamic
```

OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com
```

SSH Advanced Usage: SOCKS Proxy



PuTTY / PLink / Pageant (Windows)

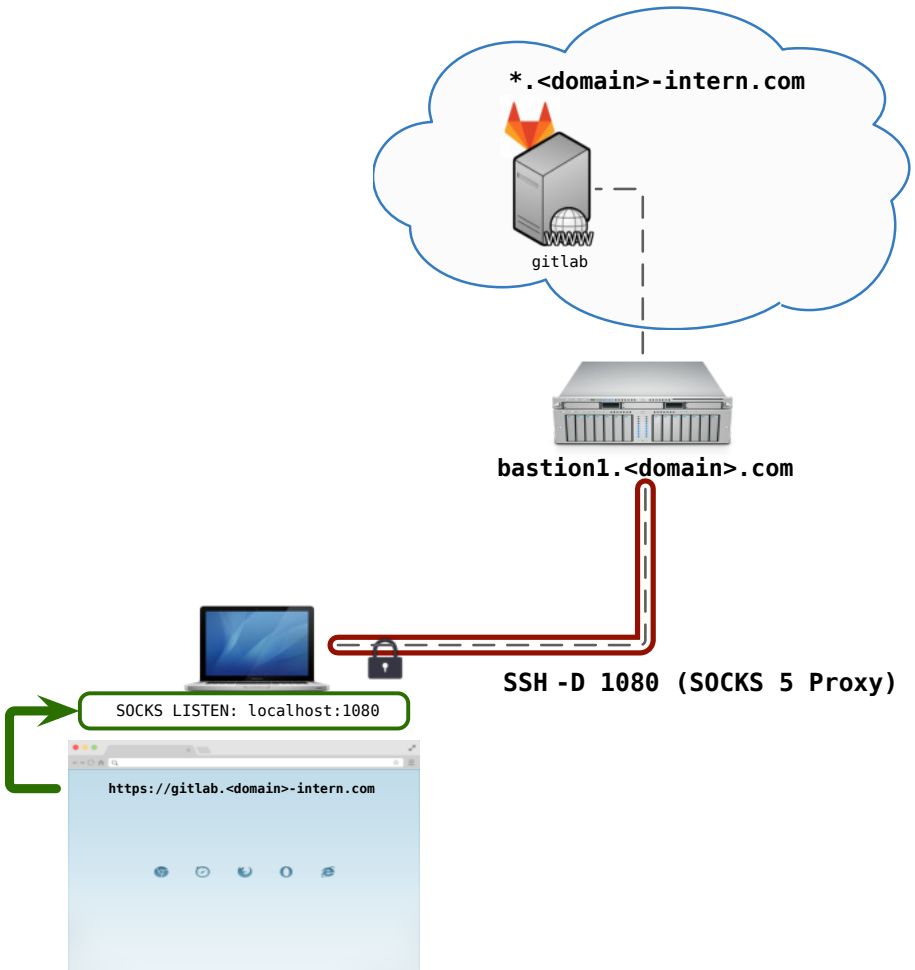
```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>
- Connection/SSH/Tunnels: Port 1080, Dynamic
```

OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com
```

SSH Advanced Usage: SOCKS Proxy



PuTTY / PLink / Pageant (Windows)

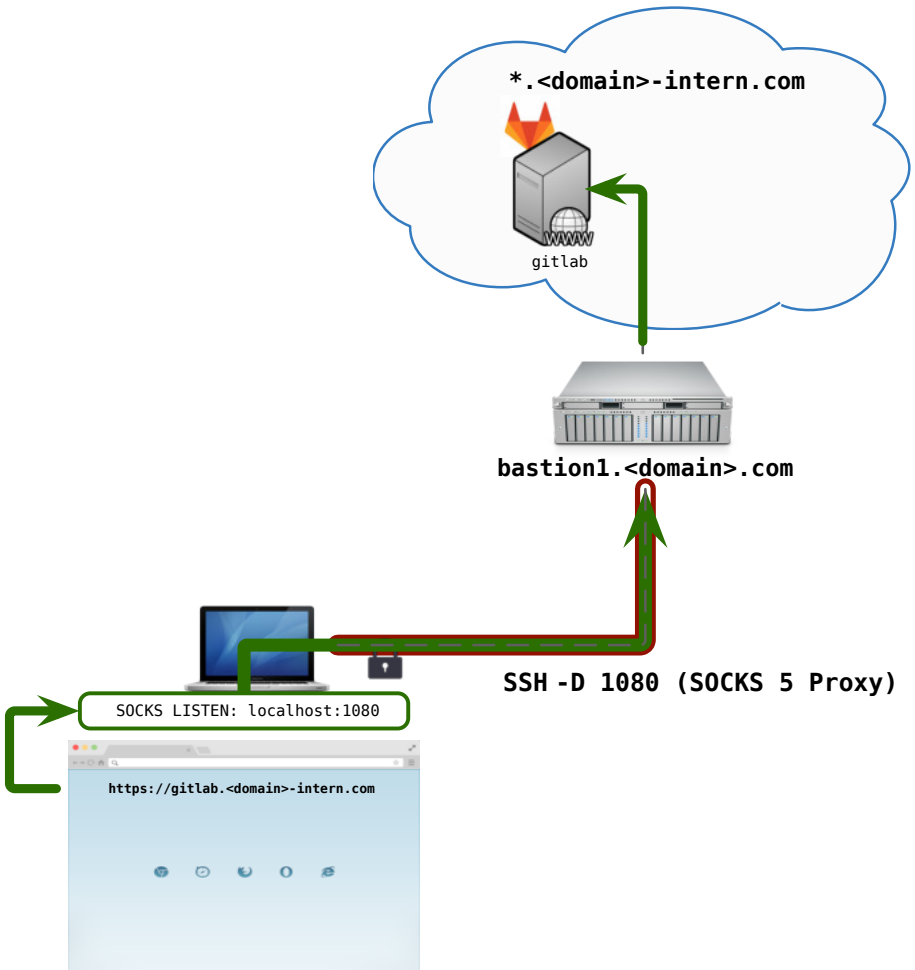
```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>
- Connection/SSH/Tunnels: Port 1080, Dynamic
```

OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com
```

SSH Advanced Usage: SOCKS Proxy



PuTTY / PLink / Pageant (Windows)

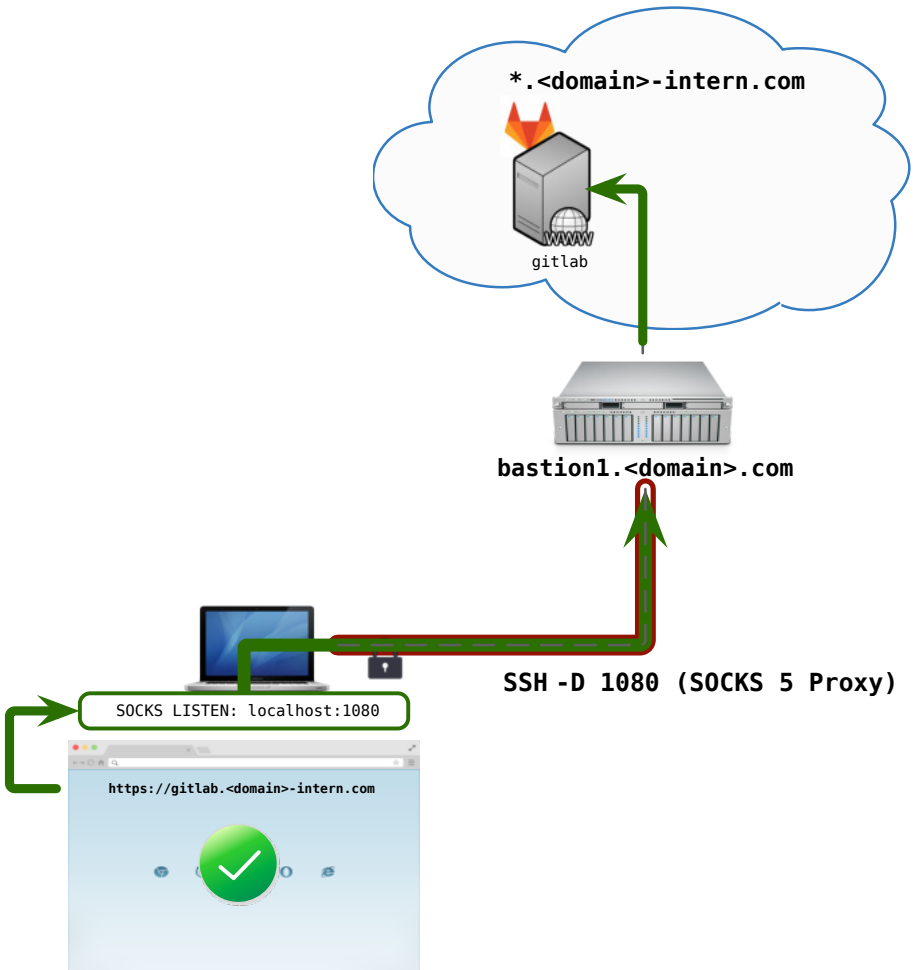
```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>
- Connection/SSH/Tunnels: Port 1080, Dynamic
```

OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com
```

SSH Advanced Usage: SOCKS Proxy



PuTTY / PLink / Pageant (Windows)

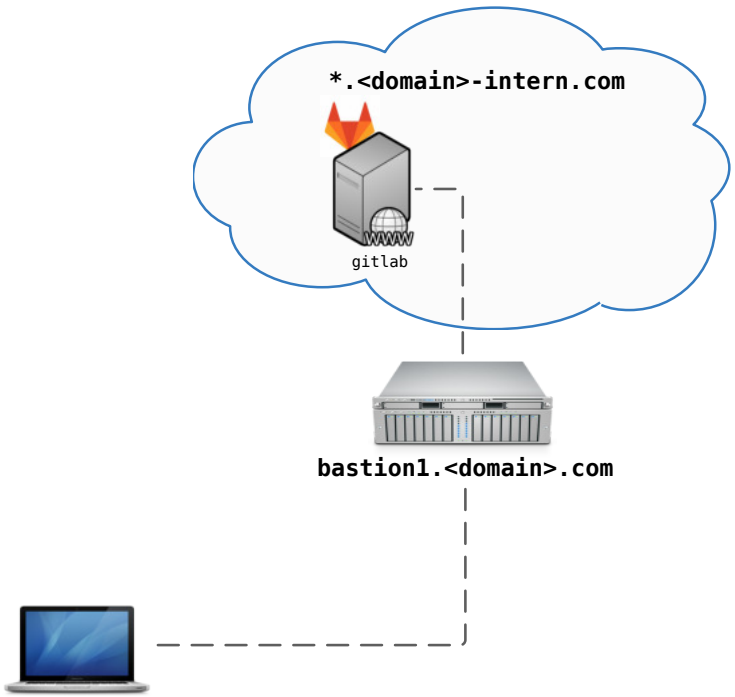
```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>
- Connection/SSH/Tunnels: Port 1080, Dynamic
```

OpenSSH ~/.ssh/config (Mac / Linux)

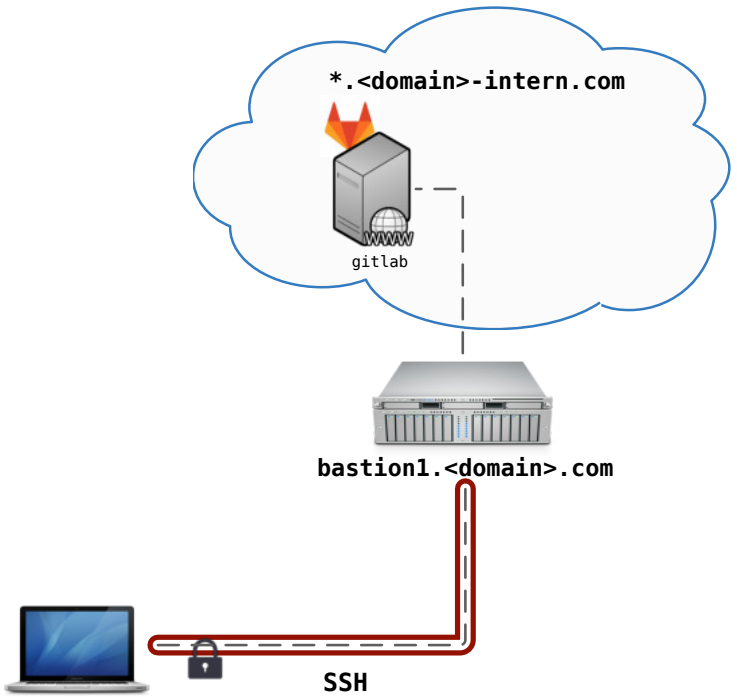
```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com
```

SSH Advanced Usage: ProxyCommand



SSH Advanced Usage: ProxyCommand



PuTTY / PLink / Pageant (Windows)

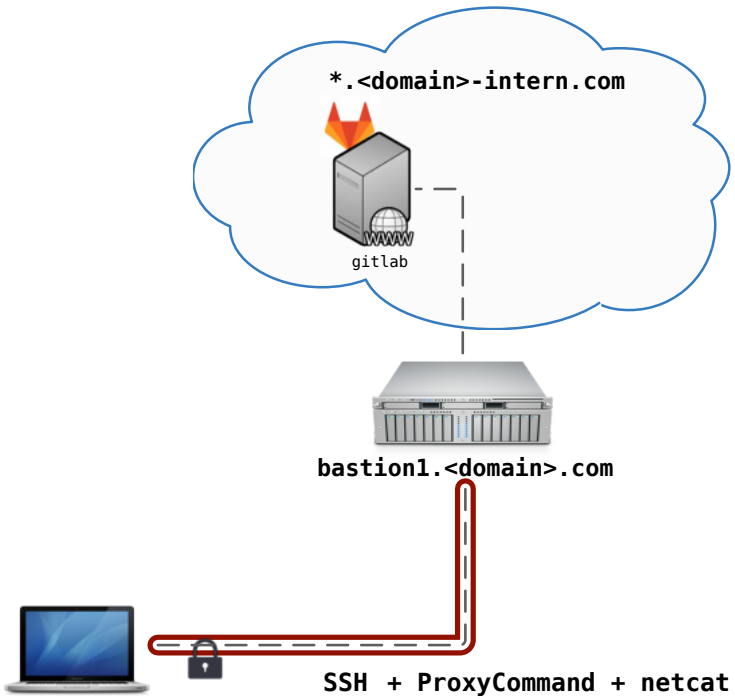
```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>
```

OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com
```

SSH Advanced Usage: ProxyCommand



PuTTY / PLink / Pageant (Windows)

```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>

Session "uni_gitlab"
- Hostname: gitlab.<domain>-intern.com
- Port: 8022
- Connection/Data: username: <LOGIN>
- Connection/Proxy:
  - type: local
  - Proxy hostname: bastion1.<domain>.com
  - Port: 8022
  - Username: <LOGIN>
  - Local proxy command:
    plink -load "uni_bastion1" -nc %host:%port
```

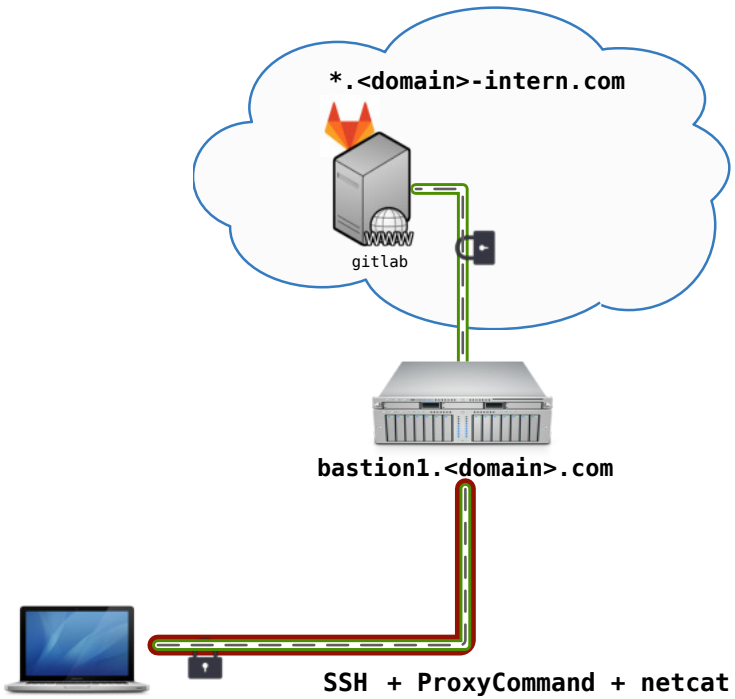
OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com

Host uni_gitlab
   Hostname gitlab
   ProxyCommand ssh -q uni_bastion1 "nc %h %p"
```

SSH Advanced Usage: ProxyCommand



PuTTY / PLink / Pageant (Windows)

```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>

Session "uni_gitlab"
- Hostname: gitlab.<domain>-intern.com
- Port: 8022
- Connection/Data: username: <LOGIN>
- Connection/Proxy:
  - type: local
  - Proxy hostname: bastion1.<domain>.com
  - Port: 8022
  - Username: <LOGIN>
  - Local proxy command:
    plink -load "uni_bastion1" -nc %host:%port
```

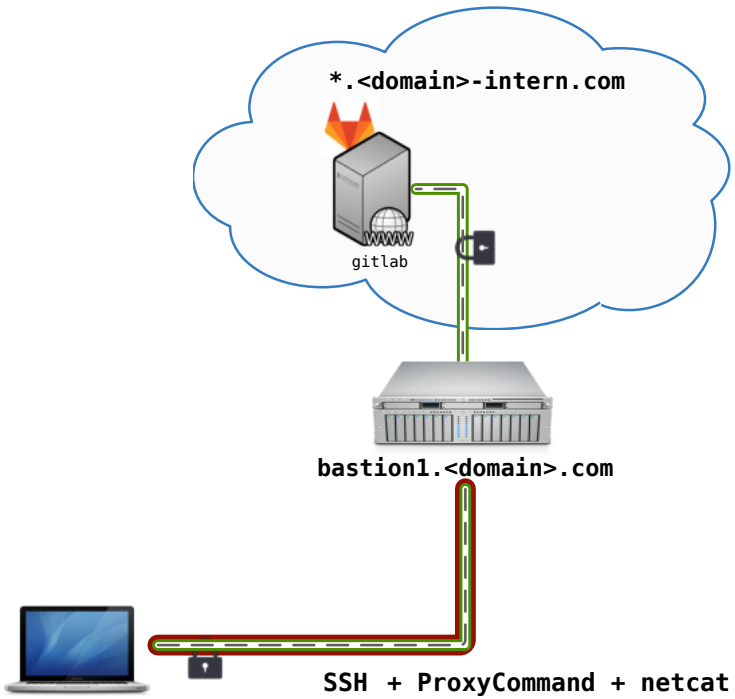
OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com

Host uni_gitlab
   Hostname gitlab
   ProxyCommand ssh -q uni_bastion1 "nc %h %p"
```

SSH Advanced Usage: ProxyCommand



PuTTY / PLink / Pageant (Windows)

```
Session "uni_bastion1"
- Hostname: bastion1.<domain>.com
- Port: 8022
- Connection/Data: username: <LOGIN>

Session "uni_gitlab"
- Hostname: gitlab.<domain>-intern.com
- Port: 8022
- Connection/Data: username: <LOGIN>
- Connection/Proxy:
  - type: local
  - Proxy hostname: bastion1.<domain>.com
  - Port: 8022
  - Username: <LOGIN>
  - Local proxy command:
    plink -load "uni_bastion1" -nc %host:%port
```

OpenSSH ~/.ssh/config (Mac / Linux)

```
Host uni_*
    User <LOGIN>
    Port 8022

Host uni_bastion1
   Hostname bastion1.<domain>.com

Host uni_gitlab
   Hostname gitlab
   ProxyCommand ssh -q uni_bastion1 "nc %h %p"
```

Summary

assh - Advanced SSH config

https://github.com/moul/advanced-ssh-config

- Transparent wrapper that make `~/.ssh/config` easier to manage
  - ↪ support for **templates**, **aliases**, **defaults**, **inheritance** etc.
  - ↪ **gateways**: transparent ssh connection chaining
  - ↪ more flexible command-line
    - ✓ **Ex**: Connect to `hosta` using `hostb` as a gateway

      `$> ssh hosta/hostb`
  - ↪ drastically simplify your SSH config
  - ↪ Linux / Mac OS **only**

```
$> { apt-get | yum | brew } install assh   # Installation
```

assh - Advanced SSH config

https://github.com/moul/advanced-ssh-config

- YAML-based **configuration**: in `~/.ssh/assh.yml`
  - ↪ use `.yml` extension, **NOT** `.yaml`
  - ↪ you can split configuration **i.e.** `~/.ssh/config.d/*.yml`
- **Hooks** support with advanced templated capabilities
  - ↪ Events: `BeforeConnect`, `OnConnect`, `OnDisconnect`
  - ↪ Exec drivers: `exec <binary> [args...]`
  - ↪ Notify driver: `notify <line:string...>`
- **Compilation**:

```
$> assh config build --ignore-known-hosts >  /.ssh/config
```

assh configuration ~/.ssh/assh.yml

```
# ~/.ssh/assh.yml - Advanced SSH Config
################################################
# Global (default) SSH flags
defaults:
  ForwardX11:      no
  ForwardAgent:    no
  ConnectTimeout: 15
  #AddKeysToAgent: yes
  Compression:     yes
  HashKnownHosts: no
  ServerAliveInterval: 60
  ServerAliveCountMax: 30
  #ControlMaster:  auto
  #ControlPath:    ~/.ssh/sockets/ssh-socket-%r-%h-%p.sock
  #ControlPersist: 600
includes:
- ~/.ssh/config.d/*.yml
- ~/.ssh/config.d/custom/*.yml
```

assh configuration: Templates

```
# ~/.ssh/config.d/templates.yml - General templates
##################################################
templates:
  # Public zone, feat. servers typically reachable from the outside
  DMZ:
    Hostname: "%h.domain.org"
    User: $USER
    Port: 22
  # internal [private] zone for the WORK domain
  WORKi:
    Inherits: DMZ
    Gateways:
    - direct        # try direct connection first...
    - gw            # ... then try through this [public] host
    - anotherserver # ... then try through this other [public] host
```

assh configuration: Hosts

```
# ~/.ssh/config.d/work.yml - ASSH config for your working place
hosts:
  #### WORK gateways / Externally accessible nodes
  mygatewayserver:    # 'ssh mygatewayserver'
    Inherits: DMZ     # eq. ssh -p 22 $USER@mygatewayserver.domain.org
    Aliases:
    - gw              # more easier to type 'ssh gw'
  anotherserver:      # 'ssh anotherserver'
    Inherits: DMZ     # eq. ssh -p 2222 $USER@anotherserver.domain.org
    Port: 2222        # custom port here
  #### WORK internal servers
  workstation:
    Inherits: WORKi
    Hostname: 10.XX.XX.XX     # if fixed IP and no DNS
    User: local
  gitlab:
    Inherits: WORKi
  storage:
    Inherits: WORKi
    IdentityFile: ~/.ssh/id_special_rsa
```

assh Basic Usage

```
$> assh config build --ignore-known-hosts >  /.ssh/config
```

assh Basic Usage

```
$> assh config build --ignore-known-hosts >  /.ssh/config
```

- Once ~/.ssh/config is compiled:

```
$> ssh <host>                                  # connect to <host>
```

```
$> ssh <host>/<gw>       # connect though (non-configured) <gw>
```

assh Basic Usage

```
$> assh config build --ignore-known-hosts >  /.ssh/config
```

- Once ~/.ssh/config is compiled:

```
$> ssh <host>                                  # connect to <host>
```

```
$> ssh <host>/<gw>        # connect though (non-configured) <gw>
```

```
$> assh connect --dry-run <host>  # dry-run verbose connection
```

assh Advanced Usage

- If you enable multiplexing / `Control{Master,Path}` settings

```
defaults:
  ControlMaster:  auto
  ControlPath:    ~/.ssh/sockets/ssh-socket-%r-%h-%p.sock
  ControlPersist: 600
```

```
$> assh sockets list              # list (opened) control sockets
```

- if you start to experience multiplexing issues:
  ↪ **Ex**: non-responding SSH connection etc.

```
$> assh sockets flush                    # Close control sockets
```

DSH - Distributed / Dancer's Shell

http://www.netfort.gr.jp/~dancer/software/dsh.html.en

- SSH wrapper that allows to run commands over multiple machines.
  ↪ Linux / Mac OS **only**

```
$> { apt-get | yum | brew } install dsh    # Installation
```

DSH - Distributed / Dancer's Shell

http://www.netfort.gr.jp/~dancer/software/dsh.html.en

- SSH wrapper that allows to run commands over multiple machines.
  - ↪ Linux / Mac OS **only**

```
$> { apt-get | yum | brew } install dsh    # Installation
```

- **Configuration**: in `~/.dsh/`
  - ↪ `~/.dsh/dsh.conf`: main configuration file
  - ↪ `~/.dsh/machines.list`: list of **all** nodes
  - ↪ `~/.dsh/group/`: holds group definition
- <name> **Group** definition: `~/.dsh/group/<name>`:
  - ↪ simply list **SSH** shortnames (one name by line)
- Bash completion file for DSH: https://gist.github.com/haron/920433

DSH configuration ~/.dsh/dsh.conf

```
#########################################################
# ~/.dsh/dsh.conf
# Configuration file for dsh (Distributed / Dancer's Shell).
# 'man dsh.conf' for details
#########################################################
verbose = 0

remoteshell      = ssh
showmachinenames = 1

# Specify 1 to make the shell wait for each individual invocation.
#    See -c and -w option for dsh(1)
waitshell        = 0  # whether to wait for execution

# Number of parallel connection to create at the same time.
#forklimit=8

remoteshellopt   = -q
```

DSH Host Group definition

- **Reminder**: in `~/.dsh/group/<name>`
  - ↪ use SSH entries as defined in `~/.ssh/config`
- Example:

```
$> cat ~/.dsh/group/iris.access
# -*- mode: conf -*-
# List of iris cluster frontends/access servers

access1.iris
access2.iris
```

DSH Host Group definition

- **Reminder**: in `~/.dsh/group/<name>`
  - ↪ use SSH entries as defined in `~/.ssh/config`
- Example:

```
$> cat ~/.dsh/group/iris.access
# -*- mode: conf -*-
# List of iris cluster frontends/access servers

access1.iris
access2.iris
```

- Additional configuration required if you wish to run sudo commands
  - ↪ **FAQ**: `sudo: sorry, you must have a tty to run sudo`
    - ✓ requires to change the default configuration of `sudo`
  - ↪ **Ex:** to **not** requiring a tty to launch a sudo command
    - ✓ typically under `/etc/sudoers.d/<login>`

      `Defaults:<login> !requiretty`

DSH Basic Usage

```
$> dsh [-c | -w] { -a | -g <group> | -m <hostname> } <command>
```

Option	Description
`-c`	run the commands in parallel (default)
`-w`	run the commands in sequential
`-a`	run the command on all nodes listed in `machines.list`
`-g <group>`	restrict the commands to the hosts group `<group>`
`-m <hostname>`	run the command only on `hostname`

```
#  pipe with 'dshbak -c' to gather the similar commands output
$> dsh -g iris.access whoami | dshbak -c
----------------
access[1-2].iris
----------------
svarrette
```

ClusterShell

https://clustershell.readthedocs.io

- ClusterShell: `clush`, `nodeset` (or cluset),
  - ↪ light, unified, robust command execution framework
  - ↪ well-suited to **ease daily administrative tasks** of Linux clusters.
    - ✓ using tools like `clush` and `nodeset`
  - ↪ **efficient, parallel, scalable command execution engine** in Python
  - ↪ provides an **unified node groups syntax** and external group access
    - ✓ see `nodeset` and the NodeSet class

```
### Installation Linux
# Debian - Ubuntu
apt-get install  clustershell

yum --enablerepo=extras install epel-release  # CentOS only
yum install clustershell    # or python34-clustershell

### OR (Mac OS...), install ClusterShell as a standard Python package
pip install --user ClusterShell
```

Configuration `/etc/clustershell/*`

- General configuration: `clush.conf`
  - ↪ Local (overriding) configs:
    `$XDG_CONFIG_HOME/clustershell/clush.conf`
  - ↪ see reference documentation - important settings:
    - ✓ `fanout` (default: 64): Size of the sliding window of ssh connectors
    - ✓ Add always all remote hosts to `known_hosts` without confirmation:
      `ssh_options: -oStrictHostKeyChecking=no`

Configuration `/etc/clustershell/*`

- General configuration: `clush.conf`
  - ↪ Local (overriding) configs:
    `$XDG_CONFIG_HOME/clustershell/clush.conf`
  - ↪ see reference documentation - important settings:
    - ✓ `fanout` (default: 64): Size of the sliding window of ssh connectors
    - ✓ Add always all remote hosts to `known_hosts` without confirmation:
      `ssh_options: -oStrictHostKeyChecking=no`

- Node groups `groups.conf`
  - ↪ defines configuration sub-directories under `groups.d`
  - ↪ cluster node groups can be defined in straightforward YAML files

YAML configuration of node group

- **Ex**: `/etc/clustershell/groups.d/iris.yaml` (see doc)

```
iris:
  access: 'access[1-2].iris'
  nodes: 'iris-[001-196]'
  gpu: 'iris-[169-186,191-196]'
  bigmem: 'iris-[187-190]'
  io: '@lustre:all,@gpfs'
  gpfs: 'gs7k1-ctrl[1-2]-vm'
  all: '@access,@compute'

# Group source cpu:
# define groups @cpu:broadwell, @cpu:skylake and @cpu:all
cpu:
  broadwell: 'iris-[001-108]'
  skylake: '@iris:nodes!@broadwell'

lustre:
  mds: 'mds[1-2].iris'
  oss: 'oss[1-2].iris'
```

ClusterShell usage: `nodeset`

- Easy manipulation of node sets, as well as node groups

```
$> nodeset [-e | -f | -c [...]]  <pattern>
```

```
$> nodeset [-l | -L]
```

Option	Description
`-c, --count`	show number of nodes in nodeset(s)
`-e, --expand`	expand nodeset(s) to separate nodes
`-f, --fold`	fold nodeset(s) (or separate nodes) into one nodeset
`-l[l]` or `-L[L]`	list groups (or all groups with -L). Double for details
...	See official doc

nodeset -e (expand) example

```
# Get list of nodes with issues
$> sinfo -R --noheader -o "%N"
iris-[005-008,017,161-162]
# ... and expand that list
$> nodeset -e iris-[005-008,017,161-162]
iris-005 iris-006 iris-007 iris-008 iris-017 iris-161 iris-162
```

nodeset -f (fold) example

```
# List nodes in IDLE state
$> sinfo -t IDLE --noheader
interactive    up    4:00:00      4   idle iris-[003-005,007]
long           up 30-00:00:0      2   idle iris-[015-016]
batch*         up 5-00:00:00      1   idle iris-134
gpu            up 5-00:00:00      9   idle iris-[170,173,175-178,181]
bigmem         up 5-00:00:00      0    n/a

# make out a synthetic list
$> sinfo -t IDLE --noheader | awk '{ print $6 }' | nodeset -f
iris-[003-005,007,015-016,134,170,173,175-178,181]

# ... actually done when restricting the column to nodelist only
$> sinfo -t IDLE --noheader -o "%N"
iris-[003-005,007,015-016,134,170,173,175-178,181]
```

Exclusion / intersection of nodeset

Option	Description
`-x <nodeset>`	**exclude** from working set `<nodeset>`
`-i <nodeset>`	**intersection** from working set with `<nodeset>`
`-X <nodeset>` (`--xor`)	elements that are in **exactly one** of the working set and `<nodeset>`

```
# Exclusion
$> nodeset -f iris-[001-010] -x iris-[003-005,007,015-016]
iris-[001-002,006,008-010]
# Intersection
$> nodeset -f iris-[001-010] -i iris-[003-005,007,015-016]
iris-[003-005,007]
# "XOR" (one occurrence only)
$> nodeset -f iris-[001-010] -x iris-006 -X iris-[005-007]
iris-[001-004,006,008-010]
```

ClusterShell usage: clush

```
$> clush [-b] { -a | -w <hostname> | -w @<group> } <command>
```

- two modes of parallel cluster commands execution:
  - ↪ **flat** mode (default): sliding window of local or remote commands
  - ↪ **tree** mode: commands propagated to the targets through a tree

Option	Description
`-b`	gathering output (as when piping to `dshbak -c`)
`-w <nodelist>`	specify remote hosts, incl. node groups with `@group` special syntax
`-g <group>`	similar to `-w @<group>`, restrict commands to the hosts group `<group>`
`--diff`	show differences between common outputs

```
$> clush -b -w @iris:access whoami
---------------
access[1-2].iris (2)
---------------
svarrette
```

Questions?

http://hpc.uni.lu

High Performance Computing @ uni.lu

Prof. Pascal Bouvry
Dr. Sebastien Varrette
Valentin Plugaru
Sarah Peter
Hyacinthe Cartiaux
Dr. Frederic Pinel
Dr. Emmanuel Kieffer
Dr. Ezhilmathi (Mathi) Krishnasamy

University of Luxembourg, Belval Campus:
Maison du Nombre, 4th floor
2, avenue de l'Université
L-4365 Esch-sur-Alzette
mail: hpc@uni.lu