Advanced SSH

Vance Kochenderfer
Western Pennsylvania Linux Users Group
July 14, 2012

Previously...

- Interactive login
  - Password authentication
  - Public key authentication
  - Host keys
- Running a single command
- Implementations
- File transfer
  - scp
  - sftp

Host Configuration Options

- Specified in /etc/ssh/sshd_config
- `Protocol` *`1|2|1,2`* - SSH protocol version(s) to support (default 2)
- `PermitRootLogin` *`value`*
  - `yes` - allow any login method (default)
  - `without-password` - don't accept password auth*
  - `forced-commands-only` - pubkey w/ `-O command`
  - `no` - root cannot log in (use `su` or `sudo`)

*Authentication methods available are GSSAPI, host-based, **public key**, **challenge-response**, **password,** and **RSA** (v1)

Host Configuration Options (2)

- Why disable root password login?
  - Opportunistic password guessing targets root
    - 26% of attempts in http://people.clarkson.edu/~owensjp/pubs/leet08.pdf
    - 57% of attempts on WPLUG server
  - No other account gets even 5% of attempts
- Can also disable certain authentication methods for all users (bold on by default)
  - `GSSAPIAuthentication` (v2)
  - `HostbasedAuthentication` (v2)
  - **`PubkeyAuthentication`** (v2)
  - **`ChallengeResponseAuthentication`**
  - **`PasswordAuthentication`**
  - `RhostsRSAAuthentication` (v1)
  - **`RSAAuthentication`** (v1)

Host Configuration Options (3)

- `Port` *`number`* - port to listen on (default 22)
  - Not really a security measure
- `ListenAddr` *`host|IP address[:port]|:port`* (default all local addresses)
- `Ciphers` *`value[,value...]`* (v2)
- `Match` *`User|Group|Host|Address value[,value...]`*
  - Can set custom options when the specified conditions are met

Client Configuration Options

- Specified on command line with -o (e.g., `-o "PubkeyAuthentication no"`), ~/.ssh/config, /etc/ssh/ssh_config
- `Protocol`, `*Authentication`, `Port`, `Ciphers` same as host options
  - Except that when multiple values are specified, they are tried in order (e.g., `Protocol 2,1` is different from `Protocol 1,2`)

Client Configuration Options (2)

- `ControlMaster` *`yes|no|ask|auto|autoask`*
  - Allows multiple ssh sessions to the same host to share a single connection
  - Also specify `ControlPath` *`pathname`*
    - e.g., `ControlPath ~/.ssh/master-%r@%h:%p`)
  - http://protempore.net/~calvins/howto/ssh-connection-sharing/

Client Configuration Options (3)

- `Host` *`pattern`*
  - Restricts following options (until another `Host` line is given) to hosts specified on command line matching pattern
  - Useful for making shortcuts to frequently-used hosts
  - If generic options desired, put a `Host *` line at end of config file followed by option specifications (remember, first value set for an option wins)

Escape Character

- Gives access to some commands while connected
- Default ~, can be changed with `EscapeChar` *`char`* or disabled with `EscapeChar none` (or `-e`)
- **Only** treated specially immediately after a newline
- Some available commands
  - Disconnect (.)
  - Suspend ssh in background (Ctrl-Z)
  - Send escape character to remote system (~)
  - List available commands (?)

X Forwarding

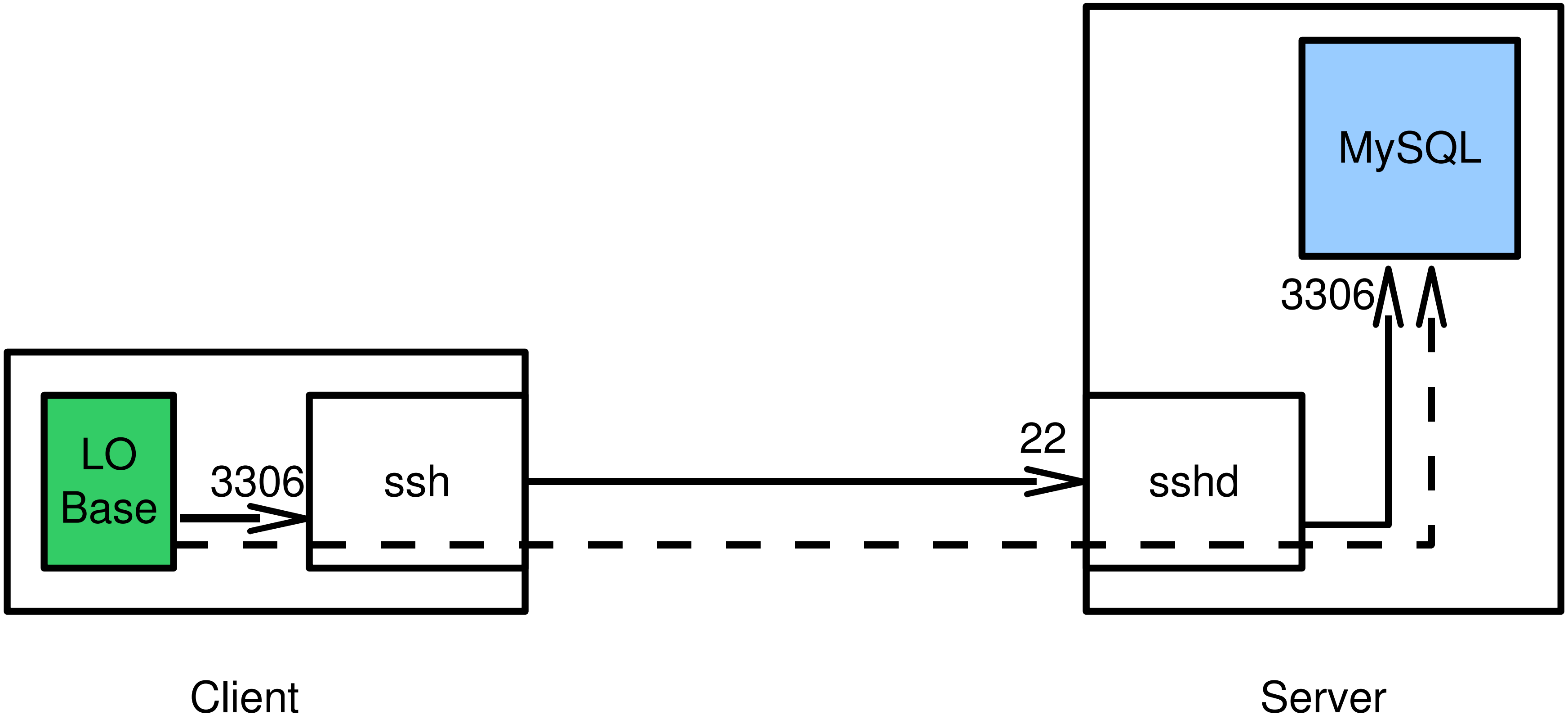
- As easy as adding `-X` to the SSH command line (or option `ForwardX11 yes`)
- Sets up fake X server on remote host which clients can connect to, `$DISPLAY` auto-set
- Using compression (`-C` or `Compression yes`) is often helpful
- X protocol not very efficient over long distances; something like NX or VNC better for frequent use

Tunneling: Local -> Remote

- `-L` *`[bind_addr:]port:host:host_port`*
  - `bind_addr` - local address to bind to (`localhost` [the default] for loopback only, `*` for all interfaces)
  - `port` - local port number to listen on
  - `host` - remote host to target (does not need to be the same machine receiving the SSH connection)
  - `host_port` - port number on remote host to target

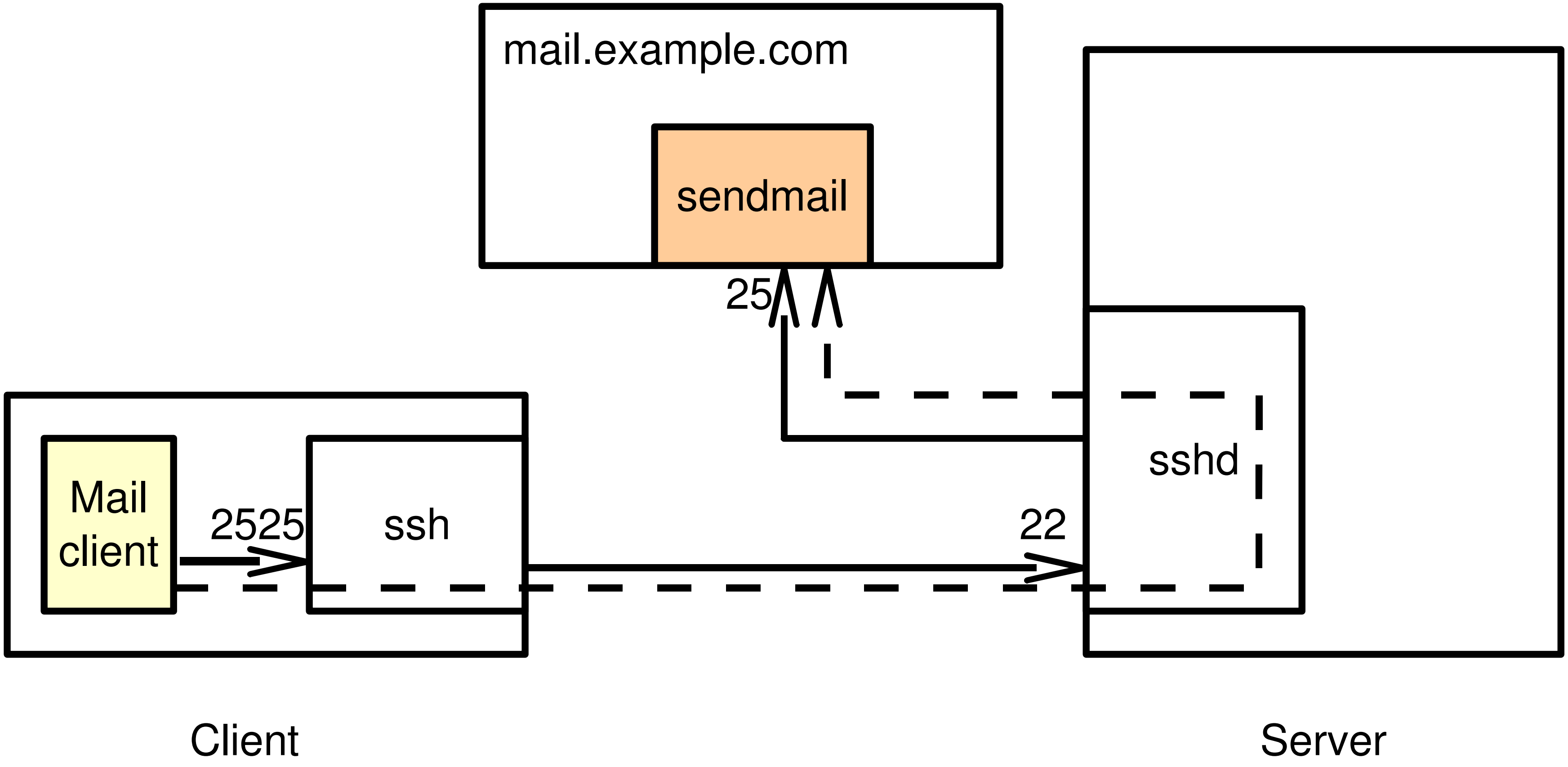
Tunneling: Local -> Remote (2)

- `-L 3306:localhost:3306`



Tunneling: Local -> Remote (3)

- `-L 2525:mail.example.com:25`

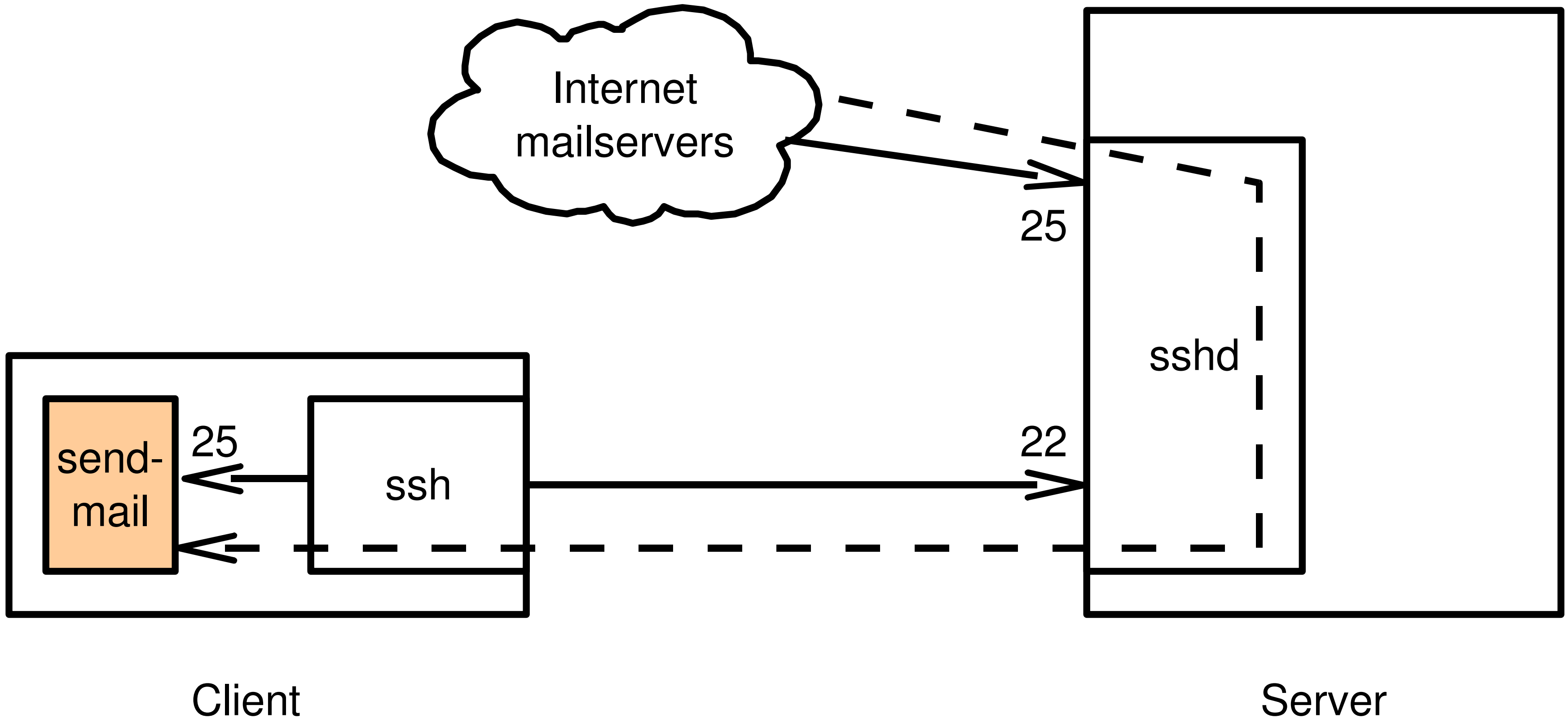


Tunneling: Remote -> Local

- `-R` *`[bind_addr:]port:host:host_port`*
  - `bind_addr` - remote address to bind to (`localhost` [the default] for loopback only, `*` for all interfaces)
  - `port` - remote port number to listen on
  - `host` - host to target (does not need to be the same machine initiating the SSH connection)
  - `host_port` - port number on target host

Tunneling: Remote -> Local (2)

- `-R 25:localhost:25`



Note: root-level access on server required to bind to port numbers under 1024

SOCKS proxy (dynamic forwarding)

- `-D` *`[bind_addr:]port`*
  - `bind_addr` - local address to bind to (`localhost` [the default] for loopback only, `*` for all interfaces)
  - `port` - local port number to listen on (1080 is IANA-assigned port for SOCKS)
- Saves having to configure port numbers
- But, applications need to support and be configured to use SOCKS