SSH-Related errors

SQLyog takes long time to connect when using SSH-tunnel.

If SQLyog is taking long time (more than 5-10 seconds) to connect to your MySQL server using SSH tunneling then there could be problem with the name resolution.
The first thing that the SSH server does before forwarding the connection to the specified MySQL server is to perform a reverse DNS lookup on the client's IP. The SSH server may cause an unnecessary delay during authentication due to incorrect or absent of reverse DNS settings. So in case of any slowness you should check those settings.
You can also disable most of the server-side lookups by setting UseDNS = "no" in SSHD configuration file (/etc/ssh/sshd_config on most systems). But in that case the MySQL host must be specified with an IP and not a hostname.

SSH - related

- "FATAL ERROR: Unable to authenticate". Occurs in case of a SSH Username/Password Mismatch or in case that you are using a username or a host IP which is listed in "DenyUsers" in the SSHD configuration file (/etc/ssh/sshd_config on most systems).
- "FATAL ERROR: Network ERROR: connection timeout". Network TimeOut (as it says) or simply could not establish contact to a SSH daemon on the URL specified. You even get this if the computer you are trying to contact is not available at all or offline. Also this error happens if you are using public/private key authentication and the keys are not valid. Keys used by SQLyog must be in .ppk -format (same as used by the 'Putty' program).
- "FATAL ERROR: Network ERROR: connection refused". Occurs typically in case of wrong SSH port. (Note that the term 'refused' implies an **activ e refusal** from the server!)
- The above error may also result from a hardware error on the remote network.
  Also those:
- FATAL ERROR: network error: Software caused connection abort.
- FATAL ERROR: Network ERROR: no route to host
  ... may be hardware related (they both can both be triggered by disconnecting the network cable). But likely that there are more reasons.
- "SSH ERROR: Unable to open connection; Host does not exist"
  ...occurs in case SSH Host does not exist (for instance in case of unsuccessful DHCP or DNS lookup - here there is no **active refusal** from the server, as there was not even an attempt to connect it!)
- **Also see** the note to the MySQL client error 2013 "Lost connection ...".

MySQL - related

As a result of this improvement we can now also retrieve more meaningful MySQL server and client errors than before - for instance:

- If MySQL user/pw is wrong result is MySQL server error 1045: "Access denied ..." as with any type of connection.
- If MySQL port is wrong result is MySQL client error 2013 "Lost connection ...".
  Note that this error also occurs if port forwarding is disabled in SSH configuration (the configuration parameter 'AllowTcpForwarding' is set to 'no' in the 'sshd_config' file). It (here) simply tells that there is no connection from SSH to MySQL for some reason. But the mySQL client API 'thinks' there was **one** connection and that is why is says 'Lost connection ...' and **not** 'Can't connect...'. There was one successful connection - but not to the MySQL server - to the SSH daemon only! But the MySQL client API is not designed to 'see' the difference!
- If MySQL port is empty or ZERO however result **is** MySQL client error 2003 "Can't connect to mysql server ..."11
- If MySQL host is wrong (or cannot be reached from SSH for some reason) result is MySQL client error 2005 "Unknown MySQL server..."

Unspecified error:

And this one is displayed if SSH connection cannot be established for some other reason and it has not been possible to resolve the error to one of the above.
"Could not establish SSH connection. Make sure that the SSH server is running and you are entering correct values for SSH port forwarding."

A concluding note on the popular 'Putty' program and SQLyog SSH-tunneling.

Sometimes when people are having problems with SSH-connections, we often hear "I can connect with Putty without problems". Maybe so, but it does not tell very much (almost nothing actually!) because a connection with Putty or a similar program **is not tunneling** and does not make use of port forwarding on the remote host. Putty simply creates a remote (and secure) shell on the client machine, and connects to the 'mysql' client program on the server. With Putty the MySQL client is the 'mysql' client on the remote server. With SQLyog SSH-tunnel the MySQL client is the client API that is compiled into SQLyog (and SJA). That is why port forwarding is needed and must be functional with SQLyog SSH-tunneling!