Alerts

Alerts are quite tricky. Basically these are fatal messages that may be passed from Client to Server or from Server to Client, which describe what went wrong to the other party, and then close the link. Now the form an alert takes depends very much on when in the protocol it is sent. So the alert may be (a) in plaintext, (b) encrypted with handshake keys, or (c) encrypted with traffic keys. If the alert is encrypted, its record type is disguised by itself being part of the encrypted payload.

When an alert is sent, it will not be received until the receiving party itself stops sending, and starts listening. A common case for sending an alert is where that which has been received simply does not make sense - a decoding error. Or if a protocol message arrives out of order. Another case would be in response to the reception of an invalid certificate chain.

Clearly an encrypted alert must be decrypted at the other end to be understood, which requires that the same keys are available at both ends for encryption and decryption.

Example

Now take for example the case where a server receives a bad certificate chain from an authenticating client. At the time the problem is identified, the server only has access to the Handshake keys. However the client having fired off its certificate chain followed by its client finished message, goes ahead and calculates the Traffic keys, and exits the protocol assuming that all has gone well. But the first record it receives from the server will be that alert, which has been waiting patiently for the client to start listening again. And the client will attempt to decrypt the alert using its Traffic keys. But the alert was encrypted with Handshake keys!

It is possible to work around this particular problem. The server must make a note that the client has failed to authenticate, but nonetheless go ahead and complete the protocol, itself calculating and switching to the traffic keys. Only then should the alert be encrypted and transmitted.

But some other scenarios cannot be fixed. For example if a bad record is received, the client and server may not be able to calculate the same keys, so the alert will decrypt as garbage.

In practise if a problem is detected at one end or the other, it may not be practical to issue an alert which has any hope of being understood by the other party. In these cases I think it makes sense to just drop the link without sending an alert.