Nested xCalls

Cross-chain calls can easily be composed together byxcall ing within thexReceive function of a target contract. In effect, the target contract becomes the source contract of that nestedxcall .

xCall in xReceive

```

Copy contractTargetisIXReceiver{ functionxReceive( bytes32_transferId, uint256_amount, address_asset, address_originSender, uint32_origin, bytesmemory_callData )externalreturns(bytesmemory) { // After handling the first xcall... ...

// Send another xcall within the xReceive function! connext.xcall{value:relayerFee}(...); } }

```

There are many ways to use nestedxcall s to extend cross-chain functionality. With this technique, it's possible to:

- Emulate the behavior of a "callback" between chains to verify state changes and/or followup asynchronously
- Disperse data to multiple different chains at once
-

See this in action in thePing Pong example.

Previous Handling Failed xCalls Next Reference Last updated9 months ago On this pageEdit on GitHub