Migrating from VRF v1

You are viewing the VRF v2 guide - Direct funding method.

To learn how to request random numbers with a subscription, see theSubscription Method guide.

Security Considerations

Be sure to review your contracts with thesecurity considerations in mind.

Comparison between VRF v1 and VRF v2 (Direct funding method)

The main similarity between VRF v1 and VRF v2 Direct funding method is that consuming contracts must be funded with LINK to pay for requests. However, Chainlink VRF v2 includes several improvements.

- Variable Callback Gas Limit:Chainlink VRF v2 lets you adjust the callback gas limit when your smart contract application receives verifiable randomness. Consuming contracts can execute more complex logic in the callback request function that receives the random values. Tasks involving the delivered randomness are handled during the response process. The new gas limits are higher than the VRF V1 limit, and vary depending on the underlying blockchain that you use. See the gas limits on theSupported networks page.
- More configuration capability:You can define how many block confirmations must pass before verifiable randomness is generated and delivered onchain when your application makes a request transaction. The range is from 3 to 200 blocks. VRF V1 always waited 10 blocks on Ethereum before delivering onchain randomness. Select a value that protects your application from block re-organizations while still providing sufficiently low latency from request to response. See theSecurity Considerations page to learn more.
- Multiple Random Outputs in a Single Request:TheVRF Wrapper contracts in VRF v2 allow you to request multiple random numbers (multi-word) in a single onchain transaction, which reduces gas costs. The fulfillment is also a single transaction, which reduces the latency of responses.

Updating your applications to use VRF v2

To modify your existing smart contract code to work with VRF v2, complete the following changes. See theGet a Random Number guide for an example.

1. Import and inherit the newVRFV2WrapperConsumerBase.solcontract and remove the v1VRFConsumerBase.solimport. This contract includes thefulfillRandomWordsfunction.
2. Add aVRFV2WrapperConsumerBaseconstructor as shown in theGet a Random Number example and use the correct VRF v2 Direct funding configuration.
3. You can still call therequestRandomnessfunction. However, the v2requestRandomnessfunction requires several different parameters (callbackGasLimit,requestConfirmations,numWords). See theSupported networks page to adjust them for your own needs.
4. ChangefulfillRandomnessfunction calls tofulfillRandomWords. Update the call to handle the returneduint256[]array instead of the singleuint256variable.