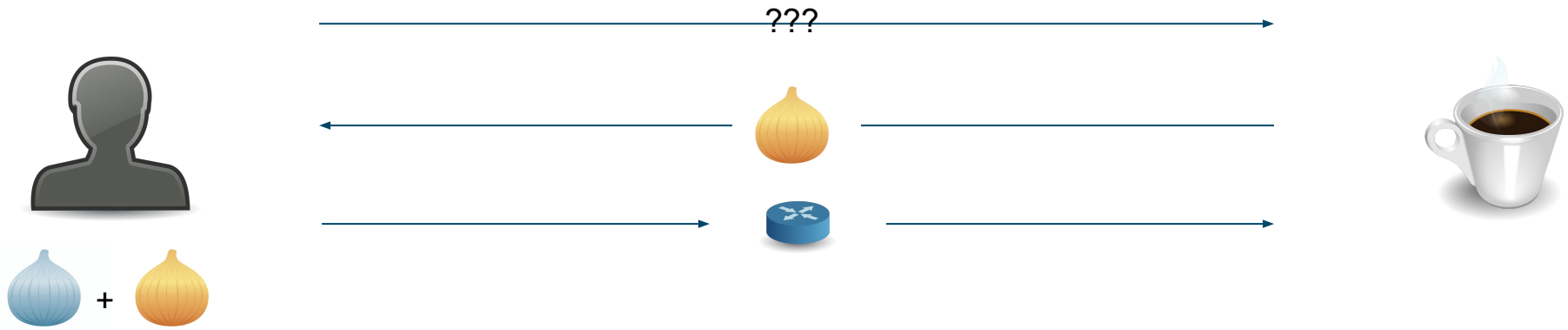
Blockstream

Rendezvous Routing

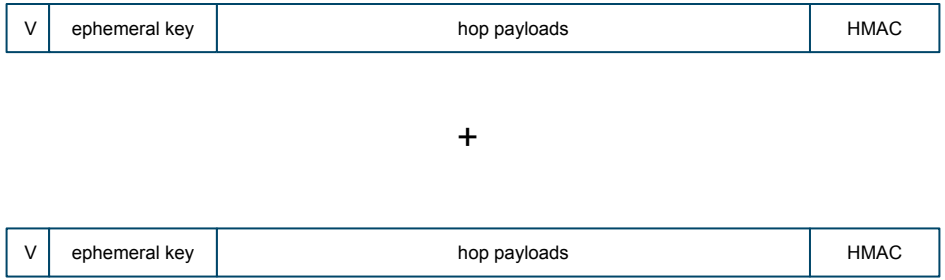
Dr. Christian Decker

Core Tech Engineer

What is Rendezvous Routing?



How do we combine two onions?



Rendez vous mechanism on top of Sphinx

Christian Decker edited this page on Nov 18, 2018 · 5 revisions

Problem Statement

We are currently using sphinx based onion routing to route a payment from one endpoint, the *sender* `S`, to another endpoint, the *recipient* `R`. This requires that the sender constructs the onion packet that contains the entire route of the payment, i.e., the sender has to compute the entire route and has to know who the recipient is.

This has a number of downsides, most importantly that the sender will learn the identity of the recipient and its surroundings. While it is possible to partially hide the recipient behind private channels and use aliases that hide the last part of the route using the route hints in the bolt11 payment requests, this approach is rather limited, as it still leaks the general location and route length.

A common feature in onion routed networks is the ability to rendez-vous node `RV` on the public network and have the recipient generate a route onion from that public location. The sender then just takes this trailing part of the route and generates a route to the public location, concatenating the two routes. This results in a route that is only partially known to either endpoint, providing anonymity to both sender and recipient.

The concatenation of the two partial routes however is not trivial, given the cryptographic primitives (EC Diffie-Hellman) used to generate shared secrets for the onion encryption. Specifically the recipient is generating an onion from `RV`, using an ephemeral key and ECDH with the hops' `node_id`. This ephemeral key is rotated on each hop using that hop's shared secret. The difficulty lies in having the ephemeral key rotated at each hop from `S` to `RV` meet up at `RV`, i.e., it being exactly the same as the one used by `R` for the `RV` to `R` route.

Proposed solution

The proposed solution is to add a new feature that allows `RV` to switch out the ephemeral key that it is passing on to the next hop, instead of rotating it via the shared secret that depends on the prior ephemeral key. `RV` receives an onion packet that contains the new ephemeral key (*FIXME: Need to describe how the presence of the ephemeral key is signaled*), and when forwarding the onion packet it'll simply prepend that ephemeral key.

http://bit.ly/lnrndvz

Thank You

@Snyke

@Blockstream

Blockstream.com