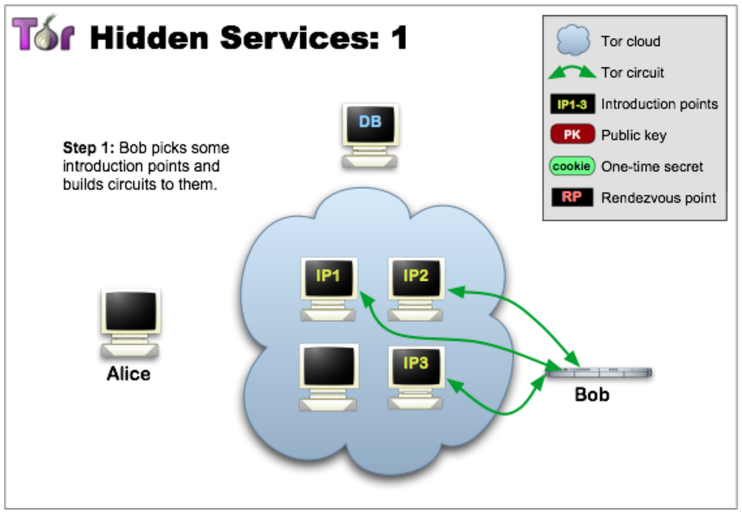
Tor Tip

Tor is written for and supported by people like you. Donate today!

Tor: Hidden Service Protocol

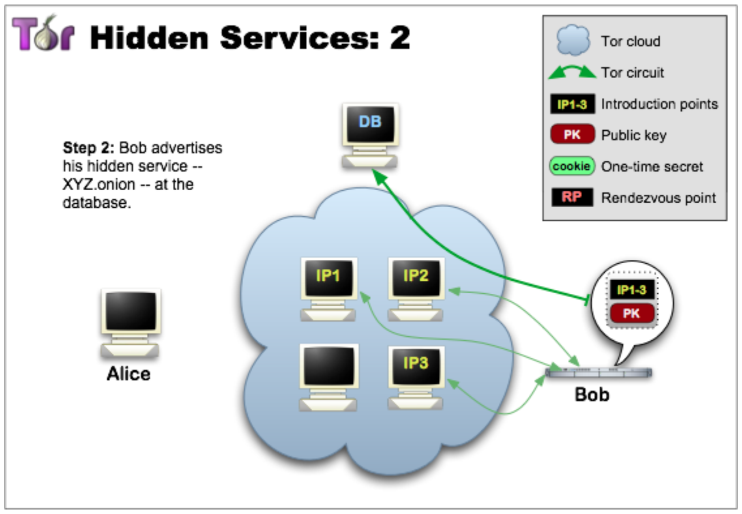
Tor makes it possible for users to hide their locations while offering various kinds of services, such as web publishing or an instant messaging server. Using Tor "rendezvous points," other Tor users can connect to these hidden services, each without knowing the other's network identity. This page describes the technical details of how this rendezvous protocol works. For a more direct how-to, see our configuring hidden services page.

A hidden service needs to advertise its existence in the Tor network before clients will be able to contact it. Therefore, the service randomly picks some relays, builds circuits to them, and asks them to act as *introduction points* by telling them its public key. Note that in the following figures the green links are circuits rather than direct connections. By using a full Tor circuit, it's hard for anyone to associate an introduction point with the hidden server's IP address. While the introduction points and others are told the hidden service's identity (public key), we don't want them to learn about the hidden server's location (IP address).

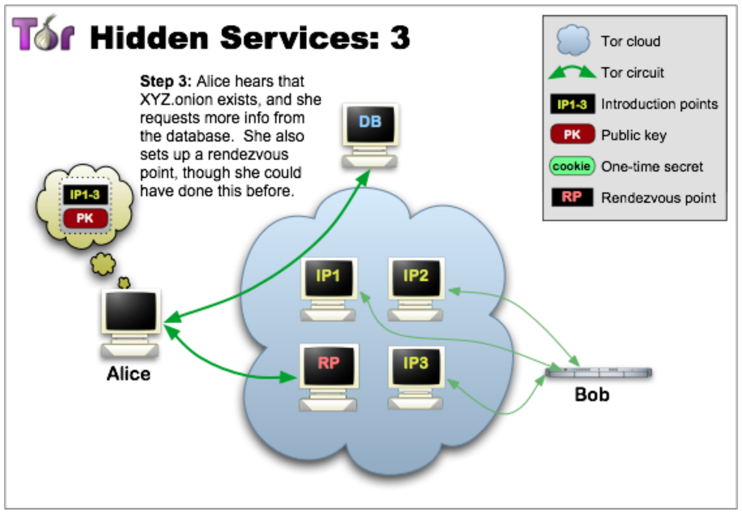


Step two: the hidden service assembles a *hidden service descriptor*, containing its public key and a summary of each introduction point, and signs this descriptor with its private key. It uploads that descriptor to a distributed hash table. The descriptor will be found by clients requesting XYZ.onion where XYZ is a 16 character name that can be uniquely derived from the service's public key. After this step, the hidden service is set up.

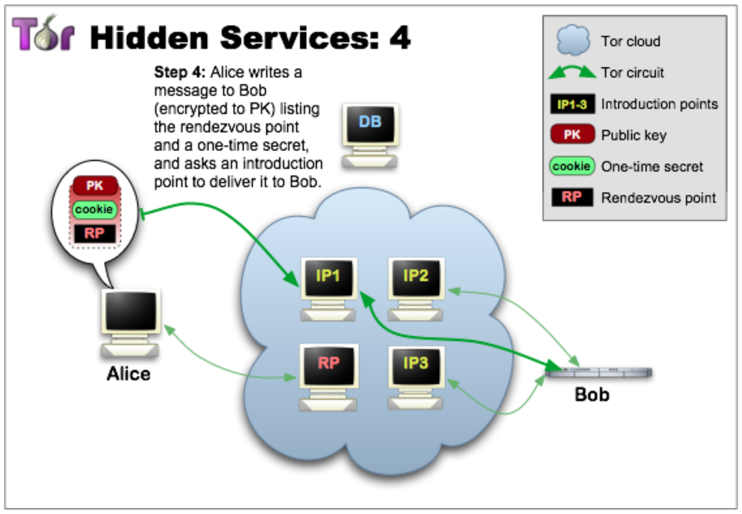
Although it might seem impractical to use an automatically-generated service name, it serves an important goal: Everyone – including the introduction points, the distributed hash table directory, and of course the clients – can verify that they are talking to the right hidden service. See also Zooko's conjecture that out of Decentralized, Secure, and Human-Meaningful, you can achieve at most two. Perhaps one day somebody will implement a Petname design for hidden service names?



Step three: A client that wants to contact a hidden service needs to learn about its onion address first. After that, the client can initiate connection establishment by downloading the descriptor from the distributed hash table. If there is a descriptor for XYZ.onion (the hidden service could also be offline or have left long ago, or there could be a typo in the onion address), the client now knows the set of introduction points and the right public key to use. Around this time, the client also creates a circuit to another randomly picked relay and asks it to act as *rendezvous point* by telling it a one-time secret.

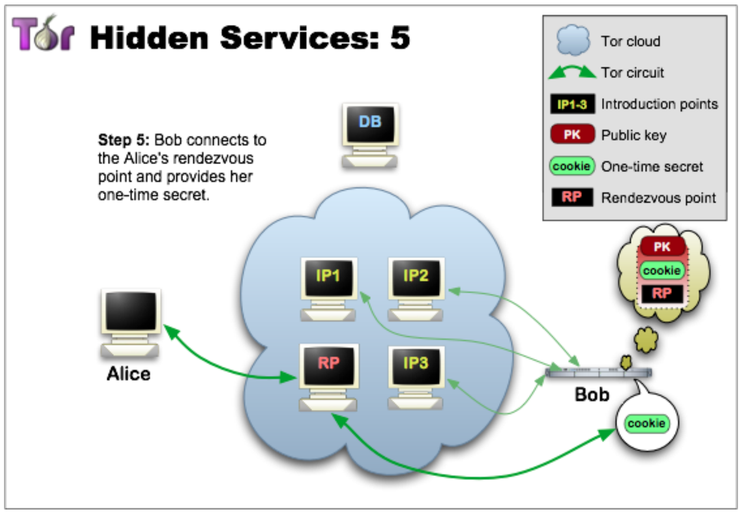

Hidden Services: 3
Tor cloud
Tor circuit
IP1-3 Introduction points
PK Public key
cookie One-time secret
RP Rendezvous point
Step 3: Alice hears that XYZ.onion exists, and she requests more info from the database. She also sets up a rendezvous point, though she could have done this before.
IP1-3
PK
DB
IP1
IP2
RP
IP3
Alice
Bob

Step four: When the descriptor is present and the rendezvous point is ready, the client assembles an *introduce* message (encrypted to the hidden service's public key) including the address of the rendezvous point and the one-time secret. The client sends this message to one of the introduction points, requesting it be delivered to the hidden service. Again, communication takes place via a Tor circuit: nobody can relate sending the introduce message to the client's IP address, so the client remains anonymous.


Hidden Services: 4
Step 4: Alice writes a message to Bob (encrypted to PK) listing the rendezvous point and a one-time secret, and asks an introduction point to deliver it to Bob.
Tor cloud
Tor circuit
IP1-3 Introduction points
PK Public key
cookie One-time secret
RP Rendezvous point
PK
cookie
RP
DB
IP1
IP2
RP
IP3
Alice
Bob

Step five: The hidden service decrypts the client's introduce message and finds the address of the rendezvous point and the one-time secret in it. The service creates a circuit to the rendezvous point and sends the one-time secret to it in a rendezvous message.

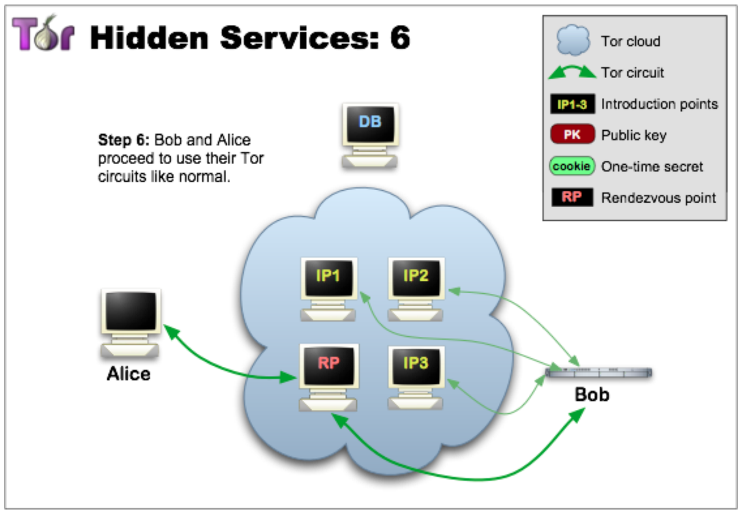
At this point it is of special importance that the hidden service sticks to the same set of entry guards when creating new circuits. Otherwise an attacker could run his own relay and force a hidden service to create an arbitrary number of circuits in the hope that the corrupt relay is picked as entry node and he learns the hidden server's IP address via timing analysis. This attack was described by Øverlier and Syverson in their paper titled Locating Hidden Servers.



In the last step, the rendezvous point notifies the client about successful connection establishment. After that, both client and hidden service can use their circuits to the rendezvous point for communicating with each other. The rendezvous point simply relays (end-to-end encrypted) messages from client to service and vice versa.

One of the reasons for not using the introduction circuit for actual communication is that no single relay should appear to be responsible for a given hidden service. This is why the rendezvous point never learns about the hidden service's identity.

In general, the complete connection between client and hidden service consists of 6 relays: 3 of them were picked by the client with the third being the rendezvous point and the other 3 were picked by the hidden service.



There are more detailed descriptions about the hidden service protocol than this one. See the Tor design paper for an in-depth design description and the rendezvous specification for the message formats.

"Tor" and the "Onion Logo" are registered trademarks of The Tor Project, Inc. Content on this site is licensed under a Creative Commons Attribution 3.0 United States License, unless otherwise noted.

About Tor
What Tor Does
Users of Tor
Core Tor People
Sponsors
Contact Us

Get Involved
Donate
Mailing Lists
Mirrors
Hidden Services
Translations

Documentation
Manuals
Installation Guides
Tor Wiki
General Tor FAQ