Running Katzenpost services behind NAT

Table of Contents

Any Katzenpost server node can be configured to support NAT and similar network topologies that traverse public and private network boundaries. This applies to directory authorities, gateways that allow clients to connect to the network, mix nodes, and service nodes that provide protocols over the mix network such as ping and spool services for storing messages or rendezvous information.

Typically, the router connecting a LAN with the Internet blocks incoming connections by default, and must be configured to forward traffic from the Internet to a destination host based on port number. These target addresses are most often drawn from RFC 6598 [https://www.rfc-editor.org/rfc/rfc6598] private address space, although more exotic topologies involving public IP address may also be targeted. (Router configuration for NAT topologies in general is beyond the scope of this topic.) For such cases, where the host listens on a LAN-side *address:port* but is accessed publicly using a different *address:port*, Katzenpost provides mechanisms to specify both addresses.

Note

Katzenpost does not support NAT penetration protocols such as NATPMP [https://www.rfc-editor.org/rfc/rfc6886], STUN [https://www.rfc-editor.org/rfc/rfc5389], and TURN [https://www.rfc-editor.org/rfc/rfc5766].

Addresses and *BindAddresses*

In a direct network connection, the values defined in the server *Addresses* parameter define the addresses on which the node listens for incoming connections, and which are advertised to other mixnet components in the PKI document. By supplying the optional *BindAddresses* parameter, you can define a second address group: LAN-side addresses that are *not* advertised in the PKI document. This is useful for NAT scenarios, which involve both public and private address spaces.

Note

The *Addresses* and *BindAddresses* parameters are closely analogous to Tor's *Address* and *ORPort* parameters. For more information, see the torrc man page [https://manpages.debian.org/testing/tor/torrc.5.en.html].

The following table shows the details for these two parameters. For more information about node configuration, see Components and configuration of the Katzenpost mixnet [https://katzenpost.network/docs/admin_guide/components.html].

Table 1. *Addresses* and *BindAddresses* parameters

Parameter	Required	Description
Addresses	Yes	Specifies a list of one or more address URIs in a format that contains the transport protocol (typically TCP), an IP address, and a port number that the node will bind to for incoming connections. This value is advertised in the PKI document.

Parameter	Required	Description
BindAd-dresses	No	If **true** (that is, if this parameter is present), this parameter sets listener *address:port* values that the server will bind to and accept connections on, but that are not advertised in the PKI document. In this case, *Addresses* defines public addresses on the Internet side of a NAT router, while *BindAddresses* defines a different set of addresses behind the NAT router.

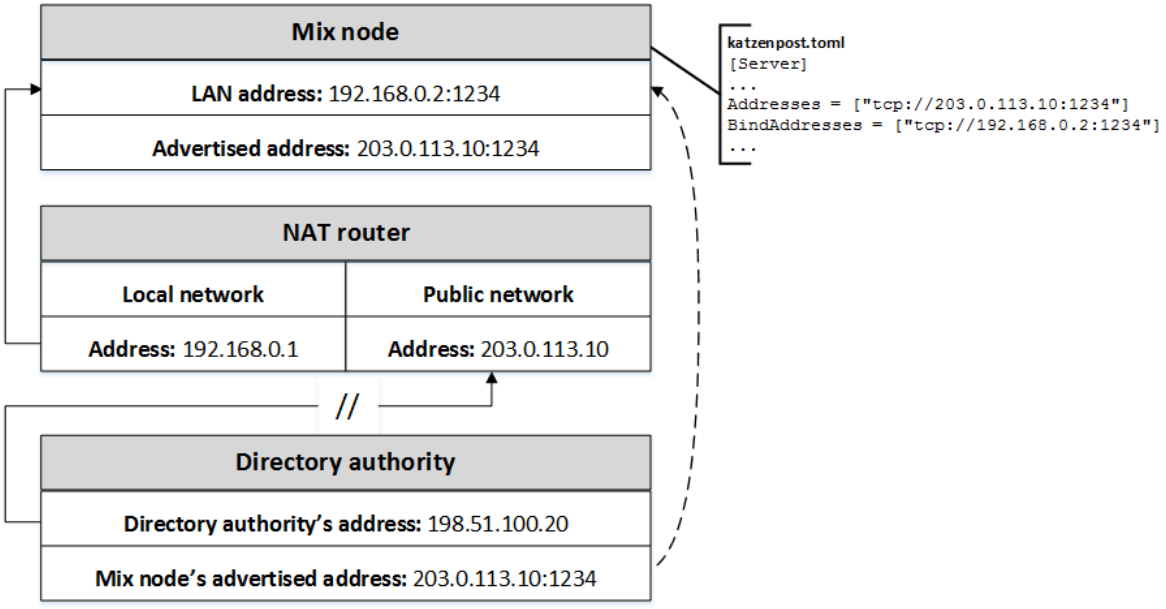
Note

Directory authorities do not support the BindAddresses parameter, but can still be used behind NAT. For more information, see Hosting a directory authority behind NAT

Hosting mix, gateway, and service nodes behind NAT

This section provides an example of a Katzenpost topology that make use of the *BindAddresses* parameter. In this scenario, a mix node behind NAT listens on local addresses for connections, while advertising a public address and port to its peer, a directory authority, that is assumed to have a publicly routable address.

Figure 1. Accessing a mix node behind NAT



Enlarge diagram [https://katzenpost.network/docs/admin_guide/pix/mix-behind-nat.png]

Key observations

- The configuration file on the NATed mix node is `katzenpost.toml`.
- The relevant section of the configuration file is `[Server]`.
- The *Addresses* parameter specifies the publicly routable *address:port*, 203.0.113.10:1234, over which the mix node can be reached from the Internet. This value is periodically advertised in the PKI document to other components of the mix network.

- The `BindAddresses` parameter specifies the LAN *address:port*, 192.168.0.2:1234, on which the node listens for incoming Sphinx packets from peers.
- The NAT router has two configured addresses, public address 203.0.113.10 and private LAN address 192.168.0.1.
- The NAT router forwards traffic for 203.0.113.10:1234 to the mix node's LAN *address:port*, 192.168.0.2:1234, where the configured listener is bound.

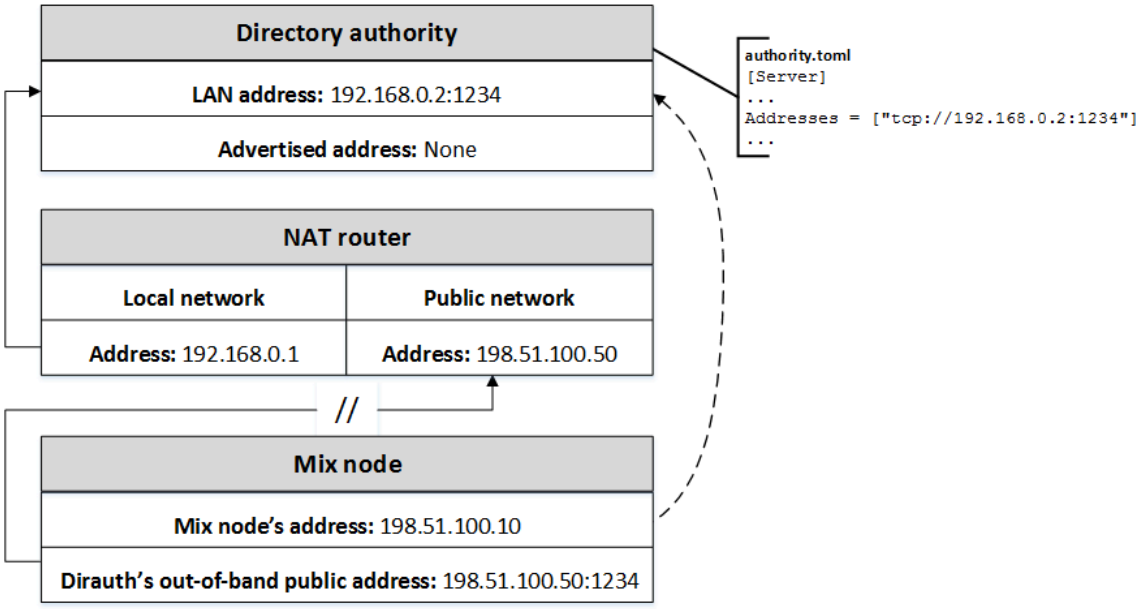
The configuration in this example applies equally well to a NATed gateway node or service provider. A NATed gateway node would also be reachable by a client with knowledge of the gateway's public address.

Hosting a directory authority behind NAT

Directory authorities have no support for the `BindAddresses` parameter. They also do not advertise an address in the PKI document, because peers must already know the address in order to fetch the document, which means that addresses for dirauths must be provided out-of-band.

Consequently, the `Addresses` parameter for dirauths performs the same function as `BindAddresses` on the other node types, that is, to define the node's listening *address:port* values, but not an advertised address. In a NAT scenario, these addresses can refer to any target that is situated on the LAN side of the NAT router.

Figure 2. Accessing a directory authority behind NAT



Enlarge diagram [https://katzenpost.network/docs/admin_guide/pix/dirauth-behind-nat.png]

Key observations

- The configuration file on the NATed dirauth is `authority.toml`.
- The relevant section of the configuration file is `[Server]`.
- The `Addresses` parameter specifies a private RFC 6598 [https://www.rfc-editor.org/rfc/rfc6598] *address:port*, 192.168.0.2:1234. By definition, this address cannot be reached directly from the Internet.
- There is no `BindAddresses` parameter.

- The NAT device has two configured addresses, public address 198.51.100.50, and LAN address 192.168.0.1.
- The NAT device routes traffic targeting 198.51.100.50:1234 to the *address:port* specified in `Addresses`, 192.168.0.2:1234.
- The dirauth does not advertise its address on the mix network. The address must provided to peers out-of-band.