

# ZeroTier Protocol

Monica Moniot

February 6, 2023

## 1 Definitions

For clarity's sake, when I use the name ZeroTier, unless otherwise specified, I am referring to the V1 ZeroTier protocol.

**Definition 1** (Node). A *node* in ZeroTier is a client machine running the ZeroTier protocol. A node is capable of making peer-to-peer, end-to-end encrypted ethernet connections with any other node, so long as it knows the address of the other node, and the other node chooses to respond to its connection. Public keys are used to identify and authenticate nodes. Currently, ZeroTier exclusively uses Curve25519 Diffie-Hellman for public and private keys.

Since all nodes are capable of communicating with each other, they form a global virtual ethernet network.

**Definition 2** (Address). The *address* of a node in ZeroTier is a uniquely identifying 40-bit number derived from the hash of the node's public key.

**Definition 3** (Identity). The *identity* of a node in ZeroTier consists of that node's public key and the matching address.

**Definition 4** (Path). A *path* is the physical link that is used for one node to send network packets to another. Generally this is a direct link over UDP, in which case the path is fully specified by the IP address and open port of the destination node. Paths are automatically found and established using DNS-like name resolution and NAT hole punching.

**Definition 5** (Network Identifier). A *network identifier* is a 64-bit number consisting of an address followed by a 16-bit "network number".

**Definition 6** (Network Controller). A *network controller* is simply a node that has had its address distributed as part of a network identifier. Network controllers are able to define a sub-network of the ZeroTier global virtual network. Nodes are members of this network if and only if they have been given permission to join it by the network controller. The network controller is also able to distribute certificates, credentials and network rules to members of its network, giving it fine control over all interactions between nodes. It is not however able to decrypt or modify traffic over its network that was not directly sent to it.

Such networks create their own virtual IP address space. When a node joins a network, it is assigned a virtual IP address on this network, and other nodes on the same network can use that IP address to contact it. The network controller decides how IP addresses are assigned, and whether to use IPv4 or IPv6. To the host machine running ZeroTier, a ZeroTier network looks and behaves as if it is a LAN or WAN that the machine is physically connected to.

**Definition 7** (MAC Address). The *MAC address* of a node in ZeroTier is a 48-bit number consisting of the address of the node xor'ed with a network identifier. As such a single node may have multiple MAC addresses but each of these MAC addresses is unique to each ZeroTier network it is a member of. These are the virtual MAC addresses assigned to machines for the sake of ethernet simulation.

**Definition 8** (Root Servers). *Root servers* are nodes operated by the ZeroTier organization that centrally handle the job of mapping all global addresses to their matching identities, and mapping addresses to possible paths to the owner of that address. Within the protocol, they are responsible for public key distribution and for coordinating initial contact between newly connecting machines.

Once keys are distributed and contact is established nodes will no longer contact root servers. The only exception being if the path connecting two nodes goes down, and a new path needs to be established, or if for whatever reason a direct path between two nodes cannot be established, so the nodes decide to route their traffic through their shared connection to the root servers.

Network traffic in ZeroTier is peer-to-peer and end-to-end-encrypted. Once a secure session is established between two nodes, it is cryptographically infeasible for the ZeroTier organization to read or modify any node's network traffic. However, it is possible for the ZeroTier organization to perform a man-in-the-middle attack on newly connecting nodes that have not yet established a session. If a root server is compromised, when it is asked to map an address to its node's identity, the root can instead brute-force the 40-bits of the address to generate an identity that matches it. It can then respond with this identity it controls instead of the correct one. The root server must then sabotage the ZeroTier rendezvous protocol so that network traffic between these two nodes is only routed through the root servers. Due to root choice randomization this attack is significantly less successful if just one uncompromised root is online. Once all this is done a classical man-in-the-middle attack can follow. Any out-of-band confirmation that the identity of the node being contacted is the same as the identity returned by the root will render this attack impossible.

Due to security and reliability concerns it is not yet possible for independent administrators to operate their own root servers. However it is possible to operate mirrors of the root servers. These mirrors, referred to as "moons" create a local copy of some subset of the root server database, and will defer to the root servers if a request is made of them for an address not present in their database.

## 2 ZeroTier Transport Protocol

To begin any connection through ZeroTier, the initiator node must send an unencrypted "hello" packet to the destination node. This hello includes the protocol version of the initiator, packet timestamp and the identity of the initiator. The destination node may decide to end the connection by simply not replying to this packet. If it does want to continue the connection and establish a secure session, it will perform Curve25519 key agreement between the initiator's identity and its own private key, generating a symmetric session key. It will then reply to the hello with a packet containing the same information about the destination node. The initiator will be able to use this to do key agreement with its private key and generate an identical session key. This session key is then used to encrypt all future packets between the two nodes. The encryption is performed using AES-GMAC-SIV with a 64-bit randomized IV. The Salsa20-Poly1305 stream cipher may also be used, in which case the 64-bit IV is used instead to mangle the session key per-packet.

Nodes will request to rekey their sessions every 60 seconds using the same protocol above. However the hello and hello reply packets are encrypted under the previous session key instead of plaintext. Rekeying requests will reset the heartbeat timer to prevent unnecessary traffic.

## 3 ZeroTier Rendezvous Protocol

All nodes start only knowing a few paths to the root servers. ZeroTier is able to bootstrap this into a peer-to-peer connection with any other node. Given the address of a node, a source node will contact a random root server to request the identity of the node with that address, the destination node. The root server will look up the destination node's identity and return it. Once the source node has its identity, it will then choose another random root server to request a rendezvous with the destination node. The root server will then search through all known paths to both the destination and the source nodes, and choose a pair most likely to succeed in back and forth communication. To the source node it will send the destination's path, and to the destination node it will send the source's path. Upon receipt both nodes will attempt to contact each other through the given paths. If either contact is successful the recipient will cache the successful path. From then on both node will choose the highest quality cached path to communicate with each other.

Upon receipt of any packet from a node, the path it was received over is either added to the path cache, or updated in the cache to help with quality determination.

## 4 ZeroTier Client Protocol

When ZeroTier is installed into a host machine, it will generate a random public and private key. The private key is saved to the host machine. The public key is used to generate an address and identity. This identity is submitted to a random root server, and if the root server confirms that the node's address is not in use, it is saved to both the host machine and the root server. If it is in use the host machine will generate a new keypair and try again.

Next the host machine will ask for a network identifier from the user. When this is given the host machine will rendezvous with the network controller for that network. Upon a successful rendezvous the node will request to join the network controller's network. If the request is approved, the network controller assigns it a virtual IP address, and sends it any network-specific credentials or rules.

ZeroTier offers ARP and NDP emulation to help client machines map a virtual IP address on a given ZeroTier network to the ZeroTier address of its node. However these protocols are not emulated 1 to 1, in particular because ARP scales poorly on large networks. Instead, when a host machine wishes to connect to a new virtual IP address, it multicasts its request to connect directly to the owner of that virtual IP address, the destination node. See [Section](#) for more details on ZeroTier multicast. If the destination node decides to connect, it will rendezvous with the host machine to establish a peer-to-peer connection. Identities and paths are always cached so in the future the host machine is able to resolve the IP address to a path to the destination node without the help of the network controller or a root server.

## 5 ZeroTier Multicast Protocol

**Definition 9** (Multicast Group). A *multicast group* is a set of nodes identified with a unique MAC address and optionally a 32-bit number, called “additional distinguishing information”, or ADI.

Nodes may send “multicast-like” packets to any other node on the same network as them. This packet contains the MAC address and optional ADI of a multicast group they would like to join. Every node locally records what multicast-likes they have received, and considers a node to be a member of a multicast group if they have sent it a multicast-like packet. Nodes automatically update known peers about their group membership status through multicast-like packets.

Nodes may send “multicast-gather” packets to any other node on the same network. This packet contains the MAC address and ADI of a multicast group. It also contains a number called the “gather limit”. A node will reply to a multicast-gather packet with a list of the addresses of every node it considers a member of that multicast group. This list will be of length at most the gather limit. No reply is sent if the list is empty.

Nodes may send “multicast-frame” packets to any other node on the same network. This packet again contains the MAC address and ADI of a multicast group, a gather limit, and a frame of arbitrary data. A node will reply to this packet the same as multicast-gather. In addition, if the receiving node is a member of the multicast group specified by the packet, the node will consider the frame of data as addressed to it and receive it.

The above protocol allows for all nodes on a given network to very quickly be updated on the members of any multicast group that the node wants to join or send frames to. This allows multicasts to be performed peer-to-peer. However, in some networks, peer-to-peer multicast is not as efficient as hub-and-spoke multicast. In those cases one may choose to use the replicator protocol for their network, described below.

**Definition 10** (Multicast Replicator). A node may become a *multicast replicator* if it has been given permission to by a network controller. Multicast replicators will replicate any multicast-frame packets they receive and forward them to every member of the multicast group specified by the packet. Nodes on a network will send their multicast frames to a multicast replicator if one exists, instead of doing sender-side replication. If multiple exist the node selects the one with the lowest path latency.

MAC address 0xffffffff is the MAC address for the “broadcast” multicast group. When any node joins a network, they automatically like this group with an ADI value of their assigned IPv4 address. This allows a multicast frame to be sent to all members of a network, or to only a member with a specific IPv4 address.

ARP emulation is performed by multicasting to group 0xffffffff with ADI the IPv4 address being looked up. IPv6 addresses in ZeroTier always contain 24-bits that identify a multicast group. Thus NDP emulation is more straightforward, as a node can multicast directly to that group to lookup that IPv6 address. If a node receives an IPv6 address it will automatically like the group identified by it.

The only access control applied to a multicast group is whether or not a node has permission to join a network. In addition, when a frame is replicated for multicast, it is the plaintext of the frame that is replicated, and then encrypted under only the peer-to-peer key of each destination node. As such the plaintext of all multicast traffic over a network is effectively fully visible to all members of the network.

## 6 ZeroTier Heartbeat Protocol

Every node creates a local database of every peer it has rendezvous with. This database stores the peer’s identity and a table of up to 64 known paths to the peer. Several quality of service metrics and timestamps are included in this database to help with making decisions throughout the ZeroTier protocol about which peers to contact or which paths to use.

Every 14 seconds, for every known path, an “echo” packet is sent over it to the peer. Nodes are expected to reply to echo packets as soon as they are received over the same path they received it over. It is when the reply to the echo is received that the node can confirm that the path is still alive. If any packet is received over a specific path, the expiry timestamp on that path is updated to the current time. If the current time and the expiry timestamp on a path ever differ by over 243 seconds, the path is considered expired and is deleted from the table of known paths.

The rendezvous protocol is specifically designed to facilitate hole punching through most NATs. However some NATs keep these holes open for very short periods of time unless the connection through the hole is actively used. The heartbeat protocol is designed to combat this and keep paths open and alive for as long as possible. This is especially important for keeping paths to the root servers alive, otherwise it would not be possible for nodes to initiate new connections.