# Staex public network

## A network for IoT devices

Staex public network is a zero-trust network that is the backbone for the todays' demand of the Internet of Things. In this paper we discuss why we are creating such a network and how it can be useful to anyone dealing with IoT devices.

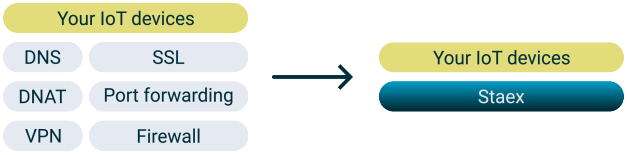
## Why public network?

Today IoT devices — IP cameras, smart meters, smart lights, smart sockets, smart fridges, etc. — are made available on the Internet through numerous cloud services, and while this solution currently works (no one can access your devices unless they hack the cloud provider) in many cases cloud does not add any meaningful functionality to what is already available through the device itself using standard protocols like HTTP, RTSP, MQTT.

The reason that we can not use this functionality without a cloud is two-fold: some of these protocols were never designed to work over the Internet and the Internet was never designed as a network for such devices. HTTP (without "S"), RTSP and many other protocols do not use encryption and public key exchange to initiate secure connection, and even if they did, bringing resource-constrained devices to the Internet will expose them to DDoS attacks and make it easy for adversaries to exploit zero-day vulnerabilities in the firmware.

The current solution to this problem (connecting these devices to the cloud) is an overkill for implementing secure communication channel. What we really need is the security on the network level so that IoT device vendors would not need to reimplement security for each of their devices. Solving this problem on a network level also improves users' privacy because they no longer need to rely on the cloud providers that often attract hackers by storing user data in one central location.

## What is Staex public network?




Staex public network is an Internet overlay that hides real IP addresses of the devices, uses public keys as addresses, and prohibits any traffic by default. Staex provides end-to-end encryption and trust on the network level, and automates making IoT devices a part of the network (port forwarding and DNAT).

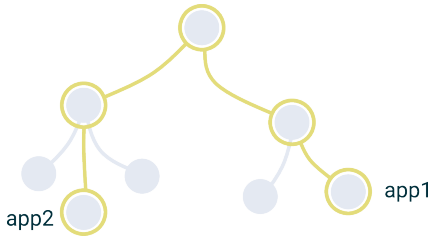
### Public keys as addresses

Staex uses the public keys of the nodes as their addresses. This approach, pioneered by Staex, aims to protect from the whole class of address spoofing attacks — the only way to spoof the address is to steal the private key, and the private key never leaves the node.

How do we map public keys to IP addresses? We use public keys as DNS names and resolve them locally on each node to dynamic IP addresses that are used to actually send IP packets.

### Hiding real IP addresses

With public keys as addresses it is straightforward to hide the real IP address: if the traffic goes through at least one intermediate node, then there is no way to know the real IP address of the device. To support such a use case Staex implements multi-hop routing: the network has tree-like topology and nodes always use shortest path between each other for the communication.



We accumulate routing tables of all the nodes in the root node. If a node can not find packet's destination in its own routing table, it forwards the packet to its parent until it reaches the root node. This means that the root node knows the topology of the whole network and can actually find the shortest path. This does not mean that the root node is different from others, in fact it runs the same program and makes routing decisions the same way as any other node. So, if you have another intermediate node between the leaf and the root and the destination node can be reached via this node, then the traffic will not reach root node at all. You can read more about the routing in the last section of this paper.

### Restricting traffic

Having node keys is not enough for the secure communication, you also need to sign them with network-wide key. Every user has such a key and this key is used to sign node keys and the packets that carry initial public key exchange data. This signature protects the traffic from man-in-the-middle attacks and relieves you from manually adding a list of authorized keys to each node (this is in contrast to how OpenSSH works by default).

Network-wide keys are also used to segment the public network into subnetworks without using the usual prefixes and network masks. By default nodes will not communicate with each other if their keys are signed by different network keys, however, you can override that by adding other network or node keys as trusted.

## Conclusion and future plans

Staex public network enables zero-trust network as a backbone of the Internet of Things. Prohibiting any communication by default and any direct communication in principle is paradoxical for something that we call "a network", however, resource-constrained IoT devices and legacy protocols can not be safely used over the modern Internet. Staex makes certificate-based trust and end-to-end encryption the default for any protocol whether it is legacy or modern.

We plan to update our built-in DNS to work properly in public network (it was designed to work in a network with single owner), and enable endpoint-based security (per-endpoint private keys and trust) with the release of Staex v2.

## Appendix

### Routing

What routing algorithm Staex network uses? Staex network topology is a directed graph. The edge goes from child node to parent node. Each node can be simultaneously a child and a parent.

The node state (e.g. its static IPv4 address, domain name, certificates) propagates from child to parent nodes, and the root of the topology — a node that does not have any parents but can reach any other node via edges — has a copy of each other node's state, i.e. knows everything about the network.

The routing uses Dijkstra algorithm that finds the shortest path between the source and destination nodes of a packet. If the path can not be found by the local node, the packet is sent to the parent node. This repeats until the root node is reached. If the root node can not find the path, then the packet is dropped.

We permit cycles in the graph, however, we encourage to stick to tree-like topologies to optimize node state propagation.

We also permit a node to have multiple parents. In this case only one parent is active at a time. When it goes offline, the child automatically switches to the next parent in the list. This allows the network to gracefully

handle root node failures and restarts.

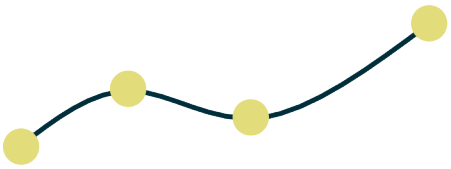
To summarize,

• we support any network topology that is a directed graph,

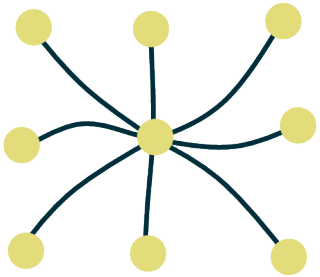
• we use shortest-path algorithm to find the best route for each packet,

• we have built-in support for root node failures that otherwise would be single point of failures.

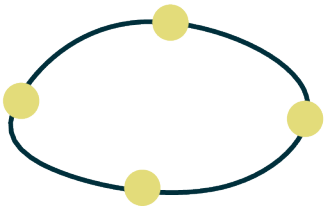
Such flexibility allows us to support use cases where some nodes have Internet connectivity, but others do not. The nodes that do not have direct access to the Internet would be able to get it via Staex network. With this multi-hop architecture it is possible to setup networks in hard-to-reach locations: forests, mountains, and even between two islands if there is enough traffic that hosts Staex nodes that work as relays.



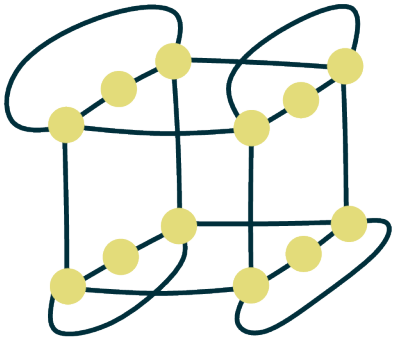
Linear



Stellar



Circular



Toroidal

staex