ngrok

Defense in Depth

May 2022

Table of Contents

2 **Introduction**

3 **Admin Dashboard**

3 **External Client Access**

4 **Authentication**

4 Mutual TLS

4 HTTP Basic Authentication

4 OAuth 2.0 and OpenID Connect

5 SAML

5 Webhook Verification

6 **Encryption**

7 Best practices for encryption

8 **Agent Configuration**

8 **Customer Data and Practices**

9 **Conclusion**

Introduction

ngrok makes it easy to secure your network traffic by providing configurable modules for authentication, encryption, and network policies. By combining components, you can meet your security requirements in a matter of minutes without rearchitecting your services.

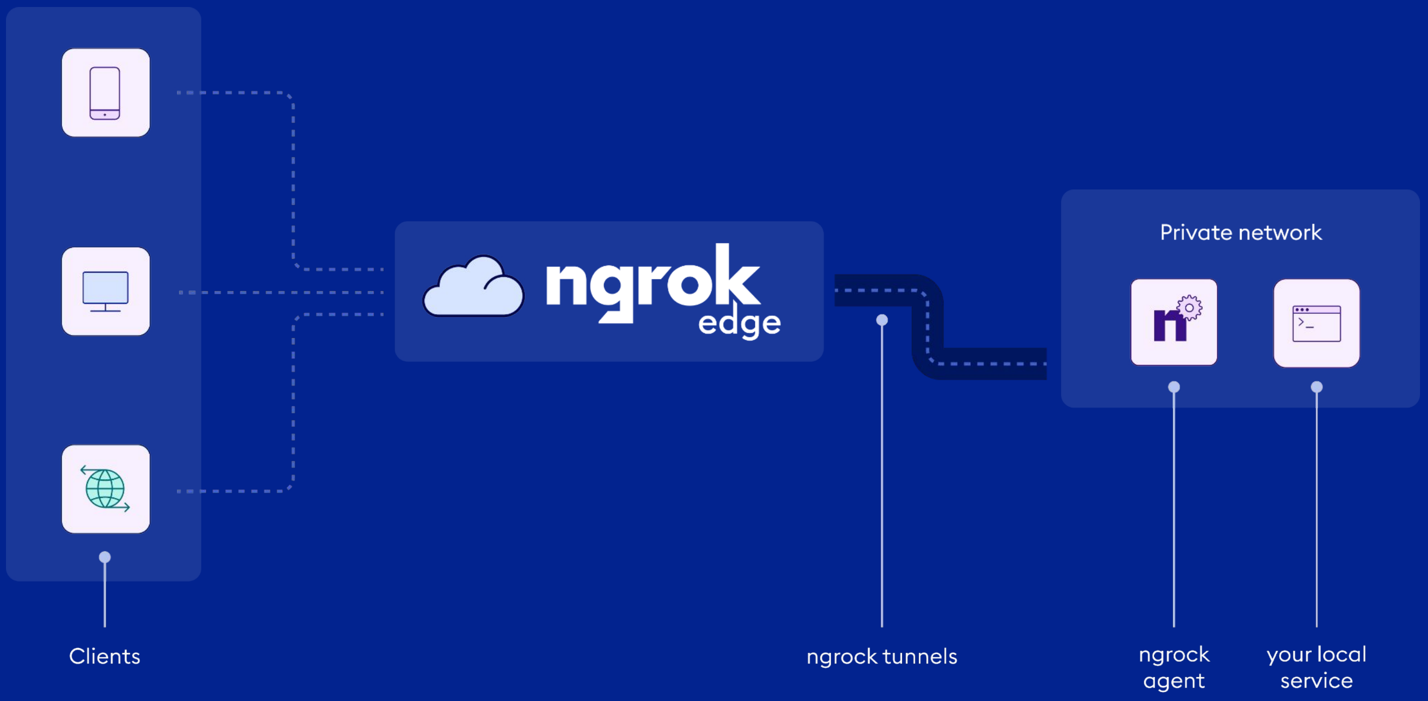


Fig 1. ngrok delivers end-to-end security within the edge traffic without rearchitecting your services

Admin Dashboard

A fundamental part of securing your services long term is ensuring their configuration is auditable by your security team. All of the security layers described later in this document are configurable and reviewable from your ngrok Dashboard. Further, the Dashboard allows Inbound Federation with OpenID Connect from GSuite or GitHub for simple integration with your existing Identity infrastructure.

External Client Access

At the outermost layer of ngrok's security options are the IP Restrictions. IP Restrictions allow you to grant or limit access to your tunnels, ngrok Agents, endpoints, the ngrok API, and even the Dashboard itself to a subset or range of IP addresses. We only recommend this approach when your expected inbound IP address range is predictable or static. Unfortunately, due to the nature of cloud architectures and distributed workforces, IP addresses are often dynamic and change frequently. In that case, with IP Policies it is still possible to use our programmatic access via our API to create, review and update IP Policies as your environment changes or even add IPv6 support.

IP Restrictions are optional but provide a first layer of protection before any other rules are evaluated.

Authentication

Once the client application meets the IP address requirements, ngrok applies authentication rules for the user or device as specified. When people consider Authentication, most think exclusively of the end-user with a username and password. While that's an option, ngrok offers both user-level and device-level approaches for a variety of customer use cases.

Mutual TLS

mTLS works by having two entities - services, devices, or even users - at opposite ends of the network, each having its own certificate. As the entities negotiate the connection, they exchange certificates to prove their own identities and create a temporary trust relationship. This ensures that the only known and expected entities can connect to your ngrok-protected service.

While this is most common in IoT devices where there is not a user present, it can also apply to the services or users of those devices just as easily.

HTTP Basic Authentication

For traditional user-based authentication, HTTP Basic Auth allows you to protect access to your service with one or more username and password combinations. This is the simplest approach for adding authentication to your service but has the drawback of a limited number of usernames and passwords so it is not suitable beyond the most basic use cases.

OAuth 2.0 and OpenID Connect

As your system grows in capability and importance, your authentication and authorization requirements will grow too. ngrok allows you to specify an OAuth 2.0 or OpenID Connect provider to extend inbound federation to protect your system. For example, we include Facebook support for consumer-oriented services or Google, Microsoft and Github for employee-related use cases.

Whichever Identity Provider you choose, ngrok will behave as any "Login with…" federated identity protected service you see on the web in general. When making a request to your service, the user will get redirected into your Identity Provider of choice, begin the authentication flow, and - upon success - get forwarded back to your service with the resulting access token.

In addition, ngrok can further refine access by enumerating certain email domains or limiting access to individual users and requiring specific OAuth scopes. This allows you to start with a general purpose Identity Provider such as Google but limit access to your organization's domain, individual users outside your domain, or both as you see fit.

SAML

For enterprise-focused use cases, ngrok supports SAML authentication. This allows you to deploy SAML over your system or service for Inbound Federation in seconds instead of weeks. ngrok also transforms some of the SAML attributes to headers for your upstream services to use.

All of these options are open standards activated via configuration options from the ngrok command-line agent or the Dashboard. To simply front and protect your service, none of these options require changing your service or underlying infrastructure. This is the most powerful approach to protect custom, legacy services with minimal modification.

Webhook Verification

Now that we've secured what can access your service via IP restrictions and who can access it via authentication, let's address the request just before it is forwarded on to your service.

Most of the major webhook providers - Stripe, Twilio, Okta, Shopify, etc - implement a verification method to ensure that incoming requests are from their trusted systems. The methods vary depending on the vendor but usually include a shared secret or certificate, a cryptographically signed payload, or both. ngrok supports many of these out of the box via command line options or the Dashboard. In limited situations, webhook vendors may publish IP ranges for their webhooks. When available, you can use those ranges with our IP Policies to further secure your endpoints.

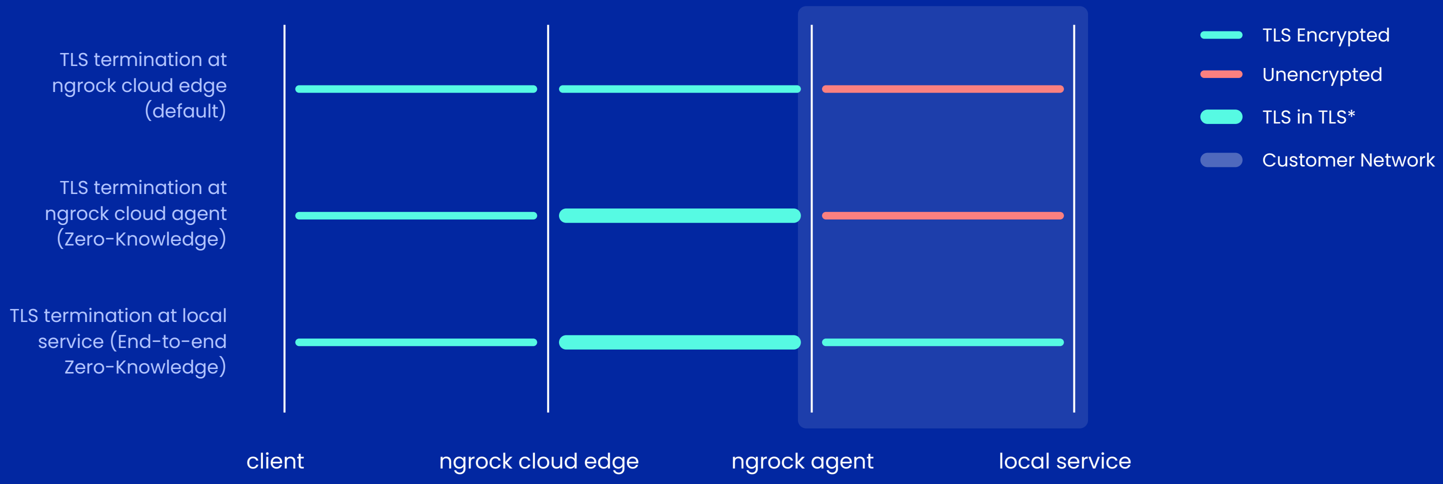
The single most important part of all of these protection capabilities - IP Restrictions, HTTP Basic Auth, OAuth, OpenID Connect, SAML, and Webhook Verification - is that they are applied at the ngrok edge long before the traffic touches your network, servers, or device. This eliminates unauthorized traffic, reduces your attack surface, and mitigates lateral attacks without modifying your service.

Encryption

Once you've used ngrok to limit who can access your service, it's important to consider the network connection between ngrok and your service. With any client to service connection through ngrok, there are three distinct connections:

1. client to ngrok cloud edge
2. ngrok cloud edge to ngrok agent
3. ngrok agent to local service

TLS Termination Options


TLS termination at ngrock cloud edge (default)
TLS termination at ngrock cloud agent (Zero-Knowledge)
TLS termination at local service (End-to-end Zero-Knowledge)
TLS Encrypted
Unencrypted
TLS in TLS*
Customer Network
client
ngrock cloud edge
ngrock agent
local service

The ngrock agent always connects via TLS. If TLS is terminated at the agent or local service, the data will be 'double' encrypted.

The protocol you need and the options you choose will determine which of those connections will be encrypted.

Best practices for encryption

While you can apply encryption to any portion of your ngrok traffic, we recommend you choose the combination which fits your use case and your organizational security requirements.

For HTTP traffic, ngrok by default provides you with best practices out of the box so you don't have to worry about it. ngrok opens only HTTPS endpoints and manages TLS certificate generation, renewal, and termination automatically. This eliminates configuration for your team while allowing our modules to handle content inspection, authentication as detailed above, and webhook verification.

If you are working in high security, high assurance industries or dealing with sensitive information, we recommend end-to-end encryption using ngrok's TLS or TCP tunnels. This requires configuring either your local service or the ngrok agent with the appropriate TLS key and certificate. It is the most secure option and helps you meet security and compliance requirements from the start but limits the additional capabilities ngrok can provide.

If your local service is not running on the same machine as the ngrok agent, we recommend that you set up TLS encryption for the 3rd leg between the ngrok agent and your local service as well.

Agent Configuration

Finally, at the innermost layer of the connection between the client and your service is the ngrok Agent running within the local network or device itself. Within the Agent, you can apply Access Control Lists (ACLs) on where ngrok Agents can be created within your network and what configurations are applied at creation. This eliminates personal ngrok accounts from your systems while ensuring allowed ngrok accounts are configured correctly every time.

Customer Data and Practices

Within ngrok itself, we collect product telemetry data and usage statistics in order to ensure uptime, defend and mitigate against abuse and fraud, and improve the system overall. The current practices and policies are detailed in our Data Processing Agreement, Privacy Policy, our Security page, and our Abuse Policy.

We also undergo third-party audit and certification through external audits such as SOC 2 Type 1 and have others on the roadmap.

Conclusion

Overall, ngrok provides a set of building blocks to allow you to configure and build the security policies your team requires. Further, by plugging into the larger ecosystem of IDPs, SIEMs, and open protocols, it fits seamlessly into your existing infrastructure.

From concept to connection, ngrok provides a secure, scalable tunnel to share your system with the world or just the single person you need.

Learn more about ngrok's capabilities here:
https://ngrok.com/product

To explore configuration options, visit
https://ngrok.com/docs