

# 1. Architecture and system requirements

ISP Go solution for Internet Service Providers consists of three components

- Filtering DNS Server
- Web application 'Block page'
- Web application 'ISP Go API' to control and manage user's settings and integrate the whole ISP Go solution with ISP's systems.

There are dependencies on the external software packages exist:

- `Nginx` is used as a reverse proxy for the block page and ISP Go API
- `Redis` is used as storage
- `Rsync` is used to download updates for domains database from SafeDNS

When a user enables a content filtering service, an ISP should set the address of the ISP Go server as the default DNS on the user's computers or devices (via DHCP, connection scripts, and so on) or forcefully redirect all DNS requests from the such user to the ISP Go server.

Using the ISP-Go API (special HTTP requests executed from the provider's scripts) this DNS server is informed about which categories of sites should not be shown to the user. The API also supports individual custom black and white lists and their global variants, which are valid for all users.

If a user types in the browser address of a prohibited site, the filtering DNS server will respond to a DNS packet from the IP address of the block page, then the browser will load the block page. The block page shows a reason why access to this domain is blocked. The block page design is customizable.

## Minimal system requirements:

The server on which the ISP-Go service can be deployed and tested with up to 100 users must meet the following minimal requirements:

- Architecture x86-64
- Debian 8, 9 or 10 amd64 installed
- 2GB RAM
- 1.6 GHz 1-core CPU
- 20 GB of free disk space
- 100 Mbit/s network card

## Recommended system requirements:

Average usage of the ISP-Go filtering service covers a few thousands requests per second. To ensure this performance should meet recommended system requirements.

- Architecture x86-64
- Debian 8, 9 or 10 amd64 installed
- 2GB RAM
- 2 GHz 4-cores CPU
- 80 GB of free disk space
- 1 Gbit/s network card

## High performance system requirements:

High performance usage of the ISP-Go filtering service covers ~0.5 million requests per second or more. To ensure this performance should meet recommended system requirements.

- Architecture x86-64
- Debian 8, 9 or 10 amd64 installed
- 8 or more GB RAM
- 2 GHz 8-cores CPU or more
- 160 GB or more of free disk space
- 2.5 Gbit/s network card or two 1 Gbit/s network cards

Currently, ISP Go is distributed as a deb-package for amd64 architecture.

For the functioning of the filtering server, an ISP should already have a standard recursive caching DNS server. **Bind9** or **Unbound** with the default settings on the other server is well suited, or you can put one of these DNS servers on the same server as ISP Go and configure it to listen only to the address 127.0.0.1.

The filtering DNS server (`isp-go-dnsproxy`) transfers all non-blocked queries to the caching server and does not cache any data itself. It is possible to pass queries of all ISP's users (even those for which the filtering service is not enabled) through the filtering DNS server without filtering or with filtering of some default categories.

The "Block page" and "ISP-Go API" web applications are designed as separate daemons, each of them listens to its own port at 127.0.0.1. To transfer requests from the external networks to these web applications, **Nginx** is used, **Nginx** listens to port 80 on the external network interface. Distribution of requests across these web applications is based on the header "Host:" in HTTP requests. All requests with a dedicated hostname for ISP Go API are transferred to the ISP Go API, while all other requests are transferred to the block page.

A provider should have any entity (billing, authentication system), which knows the correspondence between internal IP addresses and end users. A prerequisite for the implementation of the ISP Go solution for ISP is the possibility in the existing ISP's systems to run a

script when an IP address is issued to a user and (preferably) when the user logs off. Also, ISP should have a specialist who can write such scripts that will invoke SafeDNS ISP Go API on these events (user's login and log off).

A server on which `isp-go-dnsproxy` is installed should be able to send HTTP POST requests to licensing control server on the address [www.safedns.com](http://www.safedns.com). The absence of such requests is a violation of the license.

**ATTENTION!!!** `isp-go-dnsproxy` always rejects non-recursive DNS queries. This is done to eliminate the possibility of an attack like "DNS loop" when `isp-go-dnsproxy` and Bind transmit requests from one to another without end. Since Bind and other DNS servers always generate non-recursive DNS queries, browsers, and other DNS clients generate only recursive ones, then the loop will be broken. This, however, means that the configuration like "Bind with multiple zones on 127.0.0.1, `isp-go-dnsproxy` on the external address, and this external address is in NS records in these zones" is unworkable and strongly discouraged. It is not a bug but deliberately introduced limited functionality.

---

Revision #4

Created 7 September 2022 12:48:33

Updated 14 April 2023 10:48:15 by Leo Nagano