HTTP Access Service Description URIs

Ben Schwartz, MASQUE @ IETF 115

What’s an “HTTP Access Service”?

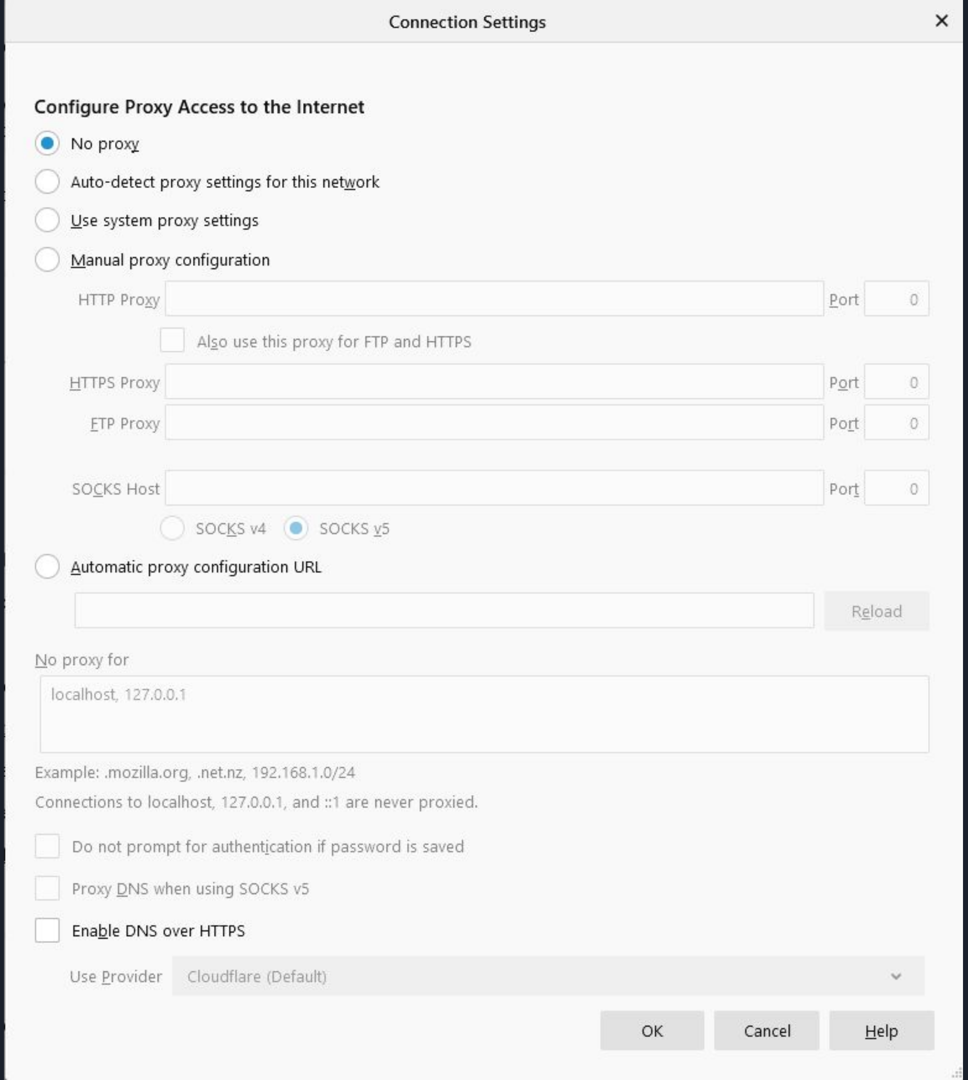
An HTTP Access Service is an HTTP server function that enables users to access to other services on the Internet.

With MASQUE, HTTP provides:

- a TCP proxy
- a UDP proxy
- a VPN server
- a DNS server
- an HTTP request proxy

How do I turn it on?

This is how you choose a proxy server



Connection Settings

Configure Proxy Access to the Internet

- [✓ No proxy]
- Auto-detect proxy settings for this network
- Use system proxy settings
- Manual proxy configuration

HTTP Proxy		Port	0
	Also use this proxy for FTP and HTTPS		
HTTPS Proxy		Port	0
FTP Proxy		Port	0
SOCKS Host		Port	0

SOCKS v4 [✓ SOCKS v5]

- Automatic proxy configuration URL

Reload

No proxy for

localhost, 127.0.0.1

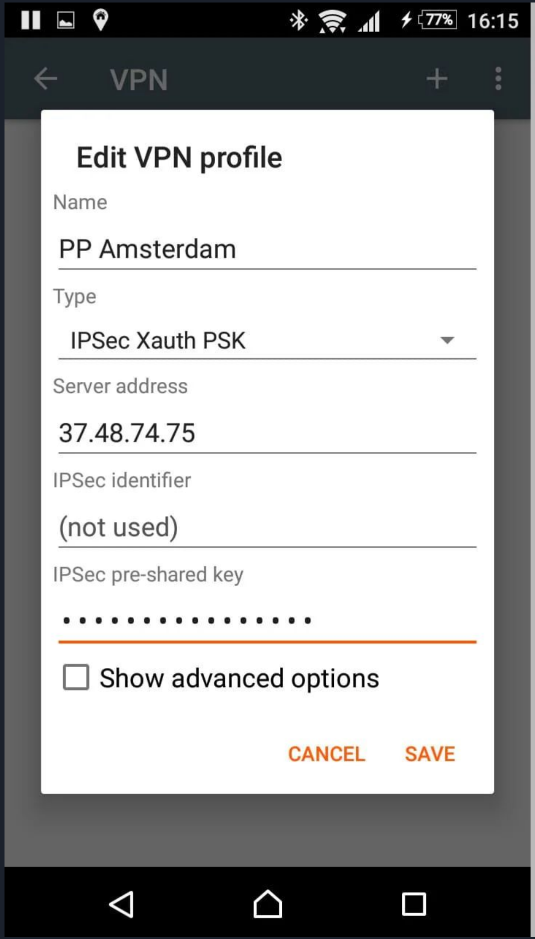
Example: .mozilla.org, .net.nz, 192.168.1.0/24

Connections to localhost, 127.0.0.1, and ::1 are never proxied.

- Do not prompt for authentication if password is saved
- Proxy DNS when using SOCKS v5
- Enable DNS over HTTPS

Use Provider: Cloudflare (Default)

This is how you configure a VPN


Edit VPN profile

Name: PP Amsterdam

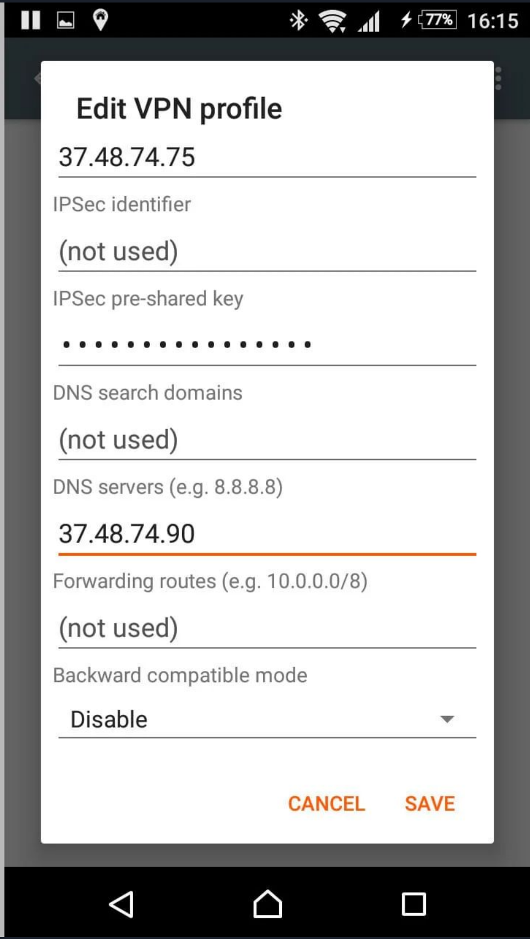
Type: IPSec Xauth PSK

Server address: 37.48.74.75

IPSec identifier: (not used)

IPSec pre-shared key: ••••••••••••••••

☐ Show advanced options


Edit VPN profile

37.48.74.75

IPSec identifier: (not used)

IPSec pre-shared key: ••••••••••••••••

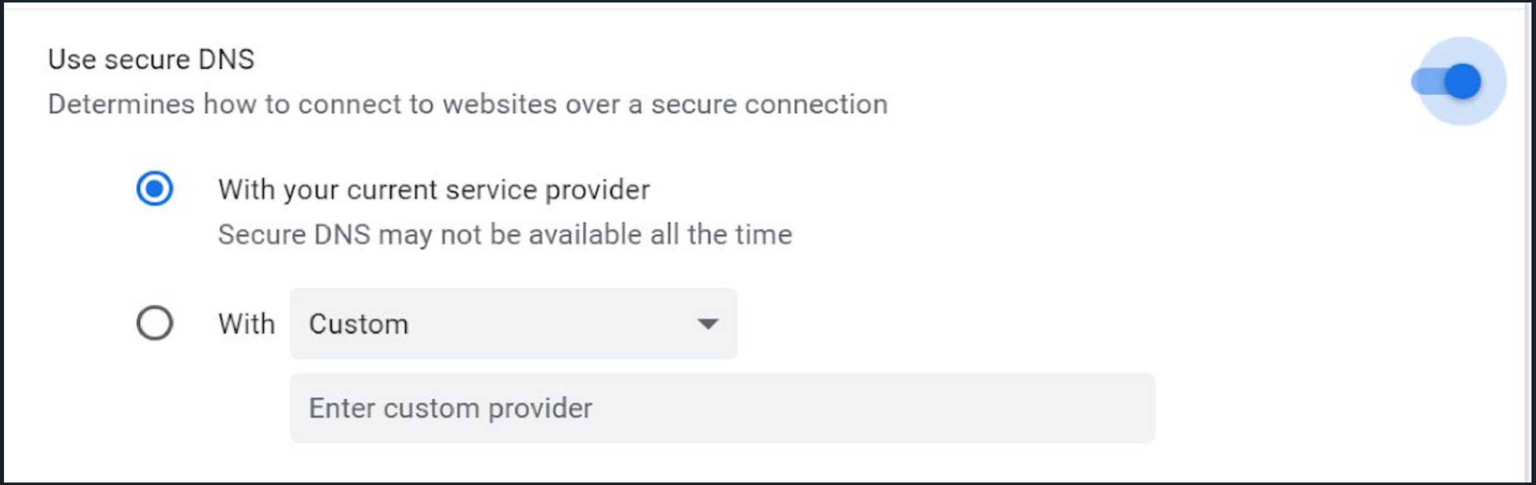
DNS search domains: (not used)

DNS servers (e.g. 8.8.8.8): 37.48.74.90

Forwarding routes (e.g. 10.0.0.0/8): (not used)

Backward compatible mode: Disable

This is how you choose a DoH server


Use secure DNS

Determines how to connect to websites over a secure connection

[✓ With your current service provider]

Secure DNS may not be available all the time

With Custom

Enter custom provider

How many times does the user need to enter this config information?

How many times does the user need to authenticate?

Goal:

- Users enter **one string** into **one UI**
- Users authenticate **once**

Proposal: Access Service Descriptions

1. The Access Service Provider gives users a URL.
2. Users paste this URL into **one place** in their client system.
3. The system tries to fetch this URL.
4. If authentication is required, the server sends "WWW-Authenticate", and the client prompts the user as needed.
5. The client retrieves the URL contents, which indicates all available Access Services.
6. The client uses these services, authenticating each HTTP request with the same credentials used for the description URL.

What’s at this URL?

```
{
  "http": {
    "template": "https://proxy.example.org/http{?target_uri}"
  },
  "tcp": {
    "template": "https://proxy.example.org/tcp{?target_host,tcp_port}"
  },
  "dns": {
    "template": "https://doh.example.com/dns-query{?dns}",
  },
  "udp": {
    "template": "https://proxy.example.org/masque{?target_host,target_port}"
  },
  "ip": {
    "template": "https://proxy.example.org/masque{?target,ip_proto}"
  }
}
```

Changes since IETF 114

- Incorporated template-driven TCP and HTTP request proxies from draft-schwartz-modern-http-proxies.
- Removed explicit OHTTP support
  - No more KeyConfigs
  - OHTTP Gateway is represented by a general HTTP request proxy
- Added discussion of authentication

Next step: **masque://** ?

A “vanity” URI scheme would enable hyperlinking Access Service Description URIs directly into the config UI, making it dramatically easier to change the Access Service configuration.

Client implementations would be responsible for handling this capability with great care!

Conclusion

- Very simple
- Reasonably future-proof
  - All keys are controlled by an IANA registry
- **Seeking adoption in MASQUE**
  - with appropriate cross-area review

Appendix

Use Cases

- Richer DNS interaction while using a proxy (without assuming a trusted third-party resolver)
  - HTTP/3 bootstrap with CONNECT-UDP
  - Encrypted ClientHello, even via “legacy” proxy configuration APIs
  - Client-side DNSSEC validation
  - “Alt-SvcB” support
- Advertising new access service features
  - e.g. CONNECT-UDP Listener mode
- Changing the capabilities of an existing service without reconfiguring the clients
  - e.g. adding CONNECT-IP support to a CONNECT-UDP proxy
- Key-Consistency DoubleCheck (related proposal in OHAI)
- Hybrid VPN + Proxy service with unified login
  - Improves performance over VPN alone

Origin vs. URL for service identification

- Access Services are identified by the URL of an Access Service Description
  - … unless this is not possible for the use case.
- If the service is identified by a hostname or HTTP Origin, we fetch /.well-known/access-services.
- We need .well-known if
  - The user already has an old-style proxy configuration, and the system wants to discover related DoH server and other modern proxy options.
  - The user knows the origin of a DoH server but doesn't know the path.
    - See ODoH example in DoubleCheck