What about Connectbyname?

Philip Homburg <philip@nlnetlabs.nl>

NLNETLABS

Connectbyname (1)

Given a host name and service, return a socket:

```
s = connectbyname(hostname, service);
```

A bit more complex, with configuration:

```
cbn_init(&cbn_ctx);
r= connectbyname(&cbn_ctx, hostname, "http", &s);
```

NLNETLABS

Connectbyname (2)

With DNS policy, and bufferevent:

```
static void callback(struct bufferevent *bev, void *ref);
static void error_cb(struct cbn_error *error, void *ref);

event_base= event_base_new();
resolver.settings=
    CBN_AUTHENTICATED_ENCRYPTION |
    CBN_DEFAULT_DISALLOW_OTHER_TRANSPORTS |
    CBN_ALLOW_DO53 | CBN_ALLOW_DOT | CBN_ALLOW_DOH2;
resolver.domain_name= "dns.example.org";
resolver.svcparams= "no-default-alpn alpn=h2 mandatory=no-default-alpn,alpn";
resolver.interface= NULL;

cbn_policy_init2(&policy, "name", 0);
cbn_policy_add_resolver(&policy, &resolver);
cbn_init2(&state.cbn_ctx, &policy, "name", 0, event_base);

r= connectbyname_asyn(&state.cbn_ctx, hostname, "https",
    callback, error_cb, &state, &ref);
r= event_base_dispatch(event_base);
```

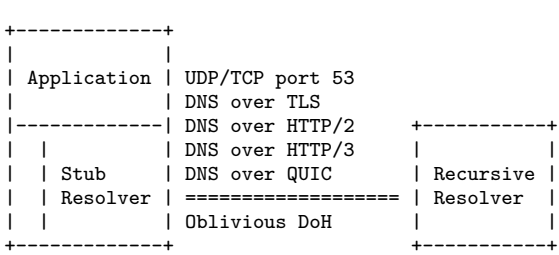
NLNETLABS

Background

- NLnet Labs worked on a connectbyname implementation under a grant from the NLnet Foundation. At the moment just a prototype.
- Asynchronous, Happy Eyeballs, DANE
- On top of getdns
- `https://nlnetlabs.nl/projects/connectbyname/about/`

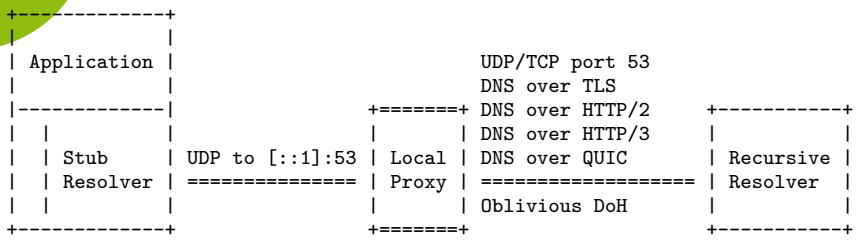
NLNETLABS

Modern Stub Resolver



NLNETLABS

Solution



A proxy on the same system as the application.

Examples of existing proxies: unbound, stubby, dnsdist, dnsmasq, systemd-resolved.

NLNETLABS

Proxy Control Option

New EDNS(0) Option

https://datatracker.ietf.org/doc/draft-homburg-dnsop-codcp/

- Stateless, send proxy control options in every request
- Maximize control of the application
- Potential for caching
- Potential for local policies in the proxy
- No DNSSEC validation requirements for the proxy
- Proof of concept:

https://github.com/getdnsapi/getdns/tree/philip-proxy-config

NLNETLABS

Apple's Network.framework (1)

From: https://developer.apple.com/videos/play/wwdc2018/715/

```
let connection = NWConnection(host: "mail.example.com", port: .imaps, using: .tls)

connection.stateUpdateHandler = { state in
    switch state {
    case .ready:
        // Handle connection establishement
    case .waiting(let error):
        // Handle connection waiting for network
    case .failed(let error):
        // Handle fatal connection error
    default:
        break
    }
}

connection.start(queue: queue)
```

NLNETLABS

Apple's Network.framework (2)

Callbacks with closures:

```
// Handle connection viability
connection.viabilityUpdateHandler = { (isViable) in
    if (!isViable) {
        // Handle connection temporarily losing connectivity
    } else {
        // Handle connection return to connectivity
    }
}

// Handle better paths
connection.betterPathUpdateHandler = { (betterPathAvailable) in
    if (betterPathAvailable) {
        // Start a new connection if migration is possible
    } else {
        // Stop any attempts to migrate
}
```

NLNETLABS

The Future is Rust (1)

Connectbyname proof of concept based on domain crate:

```
async fn do_html<IO: AsyncReadExt + AsyncWriteExt + Unpin>(mut io: IO, name: &str) {
    let req = format!("GET / HTTP/1.1\r\nHost: {}\r\n\r\n", name);
    io.write_all(req.as_bytes()).await.unwrap();

    let mut buffer = BytesMut::with_capacity(1024);
    io.read_buf(&mut buffer).await.unwrap();
    let str = str::from_utf8(&buffer).unwrap();
    println!("{}", str);
}

#[tokio::main]
async fn main() {
    let args = GlobalParamArgs::parse();
    let name = args.name;

    let mut config = Config::new();
    config.set_tls(true);
    let tls = connect(&mut config, &name, "https").await.unwrap();
    do_html(tls, &name).await;
}
```

NLNETLABS

The Future is Rust (2)

Connect function:

```
pub async fn connect<F>(config: &mut Config<'_, F>, name: &str, port_str: &str)
        -> Result<TcpOrTls, Error> where F: FnMut(bool)
```

Closure:

```
let mut better_path: bool = false;

let mut callback = |better_path_available| {
    if better_path_available {
        // Start a new connection if migration is possible
        better_path = true;
    } else {
        // Stop any attempts to migrate
    }
};

config.set_better_path_update_handler(&mut callback);
```

NLNETLABS

Feedback, Questions?

Contact me at <philip@nlnetlabs.nl>

NLNETLABS