getdns – a modern, open source DNS API implementation

Verisign Labs and NLnet Labs
https://getdnsapi.net

ccNSO TechDay, ICANN49
March 24, 2014
Introduction

• We implemented a new DNS API – what can this be used for?
  – General OS stub resolver.
  – Recursive resolution.
  – DNS resolver built into browser.
  – DNS resolver built into other apps and even servers.

• What API spec did we implement?
  – Community design by applications users needing more powerful DNS – spec published under Creative Commons CCA license, edited by Paul Hoffman, available at http://vpnc.org/getdns-api
  – DNS experts gave input and review but did not drive.
  – Goal was satisfaction of DNS user/app needs (roughly in order): standard asynchronous capability, multi-platform portability, easy upgrade to new DNS features, usability of DANE, easy general use of DNSSEC.
  – Current spec is an update capturing our implementer findings.
Make it easier for non-DNS experts to access the powerful features and evolving capabilities of DNS.
Quick Info

Verisign Labs in partnership with NLnet Labs – Strong team of Open Source developers, QA

The implementation is open source under New BSD License

We released a low-key version 0.1.0 timed with IETF 89 (2-7 Mar, 2014, London)

Release and issue management on Github:
https://github.com/getdnsapi/getdns

Website offering links, as well as binaries, documentation:
https://getdnsapi.net

– Website offering links, as well as binaries, documentation:
– Release and issue management on Github:
– We released a low-key version 0.1.0 timed with IETF 89 (2-7 Mar, 2014, London)
– The implementation is open source under New BSD License
– Strong team of Open Source developers, QA
– Verisign Labs in partnership with NLnet Labs
Some major features

- Works with a variety of event loops, each built as a separate shared library
  - Details in wiki of the github repo
  - libevent
  - libev
  - libuv
- Full DNSSEC support - base is Unbound, from NLnet Labs
- Attention to IDN handling
- Platforms as of now
  - Linux (RHEL/CentOS), MacOS, FreeBSD, iOS (now rough but usable)
  - Windows, Android later in the year
One API, two modes

• Stub resolver
  - Implemented via local library (e.g., libresolv)
  - Provides entry points for applications (e.g., gethostbyname)
  - Typically receives DNS requests via wire protocol
  - Iterates on behalf of clients or applications
  - Typically leverages caching
  - May not cache, but may implement e.g. single local cache
  - Relies on a recursive name server

• Recursive Resolver
  - Typically receives DNS requests via wire protocol
  - Iterates on behalf of clients or applications
  - Typically leverages caching
  - May not cache, but may implement e.g. single local cache
  - Relies on a recursive name server

Getdns- api context controls which of these (2 modes)
DNSSEC validation is off by default for stub mode (by design)

- dnssec-return-validation-chain
- dnssec-return-only-secure
- dnssec-return-status

Defined in API

Group consensus), but easy to turn on – use of extensions

The API spec allows enabling DNSSEC on a per-request basis
via setting the dnssec-return-status extension. For
convenience, the implementation provides a means to
enable this extension for every request in a given context

Documented in getdnsapi repo community wiki
This API is not familiar to C/sysadmin/DNS programmers, but it matches style of web/apps programmers.

This API is not familiar to C/sysadmin/DNS programmers.

Getdns-General is typical of public entry points but it matches style of web/apps programmers.

Getdns-General handles arbitrary resource record types.

API examples – Getdns-General()
API examples - getdns_address()

- Handles requests by host name
- Always returns both IPv4 and IPv6 addresses
- Uses all name spaces from the context

```
getdns_return_t getdns_address(getdns_context_t context, const char *name, struct getdns_dict *extensions, const char *name, const getdns_context_t context, getdns_return_t)
```
API examples -

- getdns_hostname

  • Accepts either IPv4 or IPv6 address

AP1 examples - getdns_hostname()
API examples - getdns-service

getdns_service

getdns_service

Returns relevant SRV information

API examples - getdns-service()
Reac9on'and'Plans'

• Immediate interest from open source and IETF communities following the 0.1.0 release
  Node.js bindings, Python bindings due around 11 April
  Release 0.1.1 that comes out this week will include
  Particular interest in Node.js and Python bindings
  Node.js bindings, Python bindings...
  Akamai, Facebook, Cisco, Mozilla, ...
  Supportive statements at IETF from folks from Microsoft,
  Mac Homebrew formula has appeared
  Received a patch for RHEL packaging the day of release
  Multiple forks of the getdns and getdns-ios repos already

• Upcoming trials in web developers’ hackathon (at TNW)

Reaction and Plans
Still to be implemented

• These items are in the API spec and will appear in
  subsequent release

  • Full list in README
  • ...

  • Full set of EDNS(0) and OPT extensions
  • GETDNS-TRANSPORT-TCP-ONLY-KEEP-CONNECTIONS-OPEN
  • MDNS and NetBIOS namespaces – included in spec
  • DNS search suffixes – Getdns-context-set-append-name,
    Getdns-context-set-suffix – followwing DNSOP discussions...

Still to be implemented
Implementation Detail - Dependencies

- libidn from FSF, version 1
- libexpat
- (headers and libraries)
- libidn and libunbound from Nlnet Labs (libidn requires openssl)
- Current set
- We strive to minimize them
- They are linked outside the build tree, with configure finding
The Getdns core team

- Glen Wiley
- Wouter Wijngaards
- Duane Wessels
- Willem Toorop
- Melinda Shore
- Allison Mankin
- Olat Kolkman
- Neel Goyal
- Angélique Finan
- Craig Despaux
Questions?

Most answers will be found at

getdnsapi.net

github.com/getdnsapi