

DNSSEC for Legacy Applications - POC

Presenter: **Sara Dickinson (Sinodun)**

Allison Mankin, Gowri Visweswaran (Verisign Labs)

Theogene H. Bucuti (University of North Texas)

Willem Toorop (NLnet Labs)

ICANN54 DNSSEC Workshop - October 21, 2015

Goal of this work

- Enable all applications to benefit from the security provided by DNSSEC and access new DNS features such as privacy

Background

- Linux and UNIX systems provide a default DNS resolver library
 - Application name resolution via `getaddrinfo()`, `getnameinfo()`, etc.
 - The majority of applications use the system resolver
 - Also, some applications (such as browsers) may use their own resolver library
- ISSUE: Current library implementations do not support DNSSEC nor other modern DNS capabilities

Background

- A DNSSEC validating recursive could provide secure DNS
 - However the 'last mile' issue is unsolved
 - Does not enable new features such as privacy
- Libraries such as **getDNS** allow applications to consume these features [DNSSEC, TLS] (<https://getdnsapi.net/>)
 - However this requires making changes to the application

Background - more details

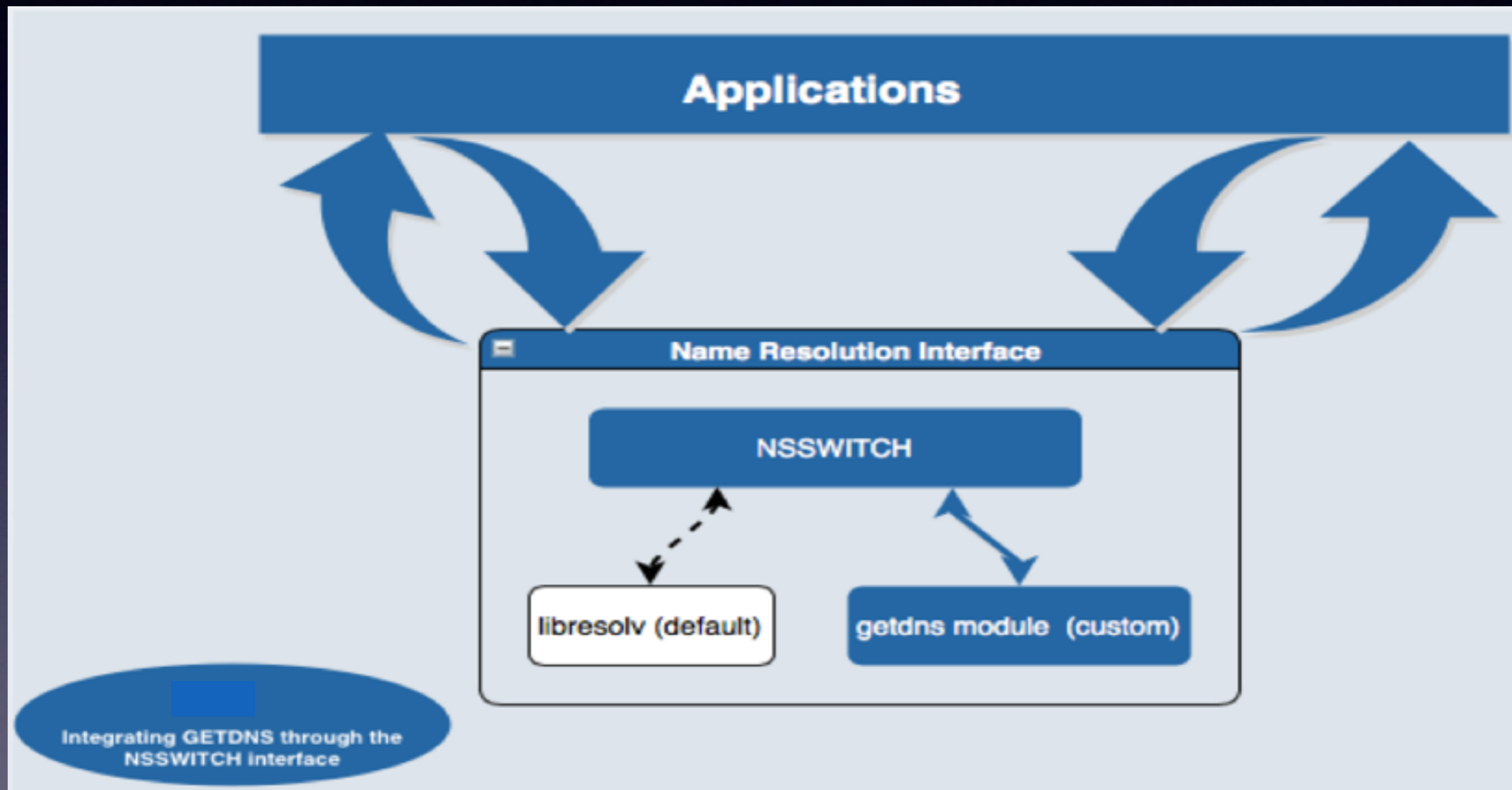
- **Name Service Switch (NSS)** - Linux and UNIX service that provides pluggable interface for system services
- Administrators configure which 'services' provide information to the runtime libraries (e.g. users, name resolution)
 - Name resolution services are implemented by shared object libraries and configured in `/etc/nsswitch.conf`
 - Default config: `'/etc/hosts'` then DNS

```
hosts:    files    dns
```

Goal of this work

- Enable all applications to benefit from the security provided by DNSSEC and access new DNS features such as privacy
- **Solution:** A new NSS service that uses *getDNS* to provide these features
- By using NSS this will be transparent to applications
 - Any application that uses the standard system API will seamlessly get support for DNSSEC.

Proof of Concept (POC) Architecture



- *getdns* in validating stub mode
- Experimented with LD_PRELOAD as well

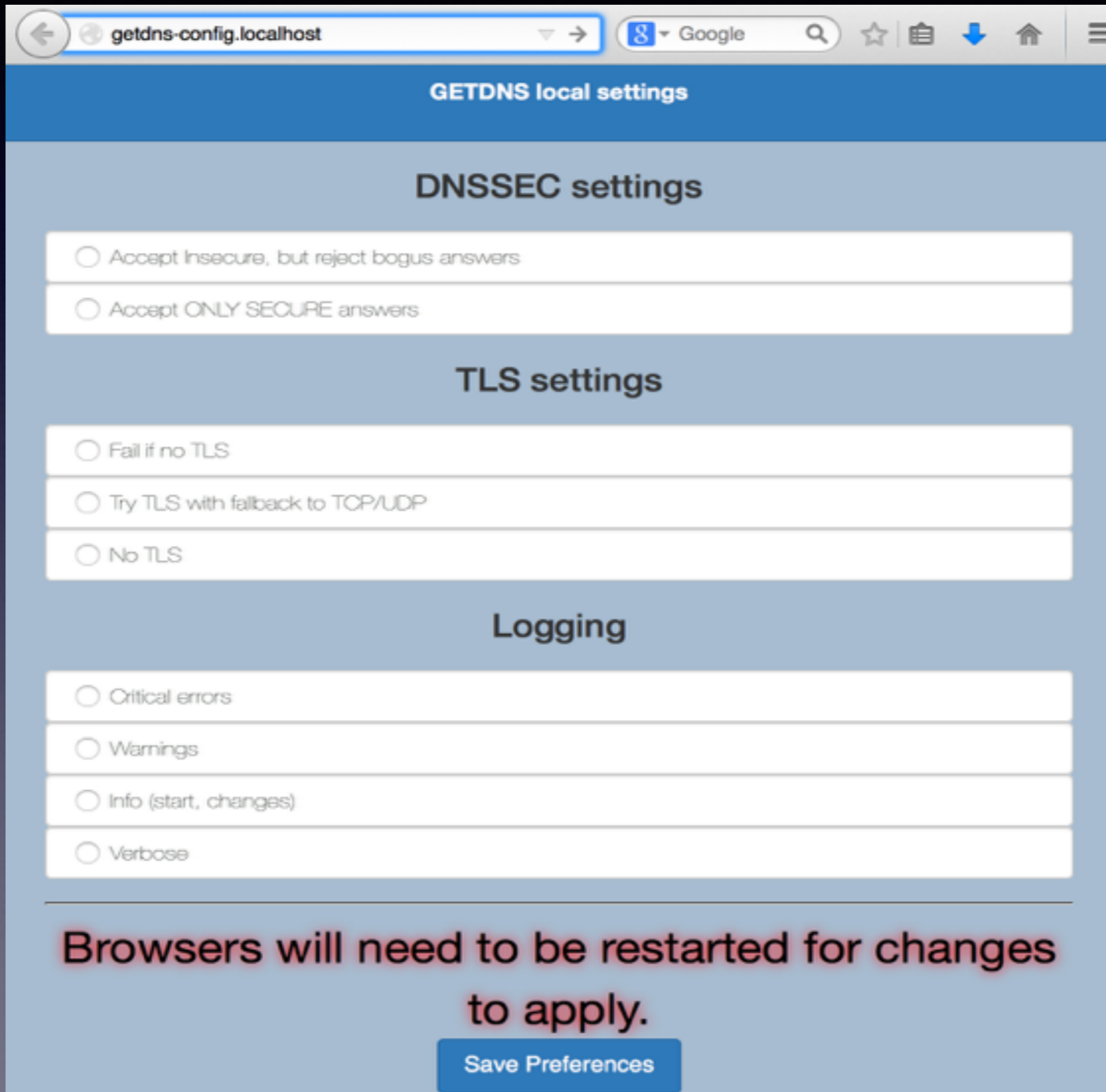
Configuration

| ACTION | FILE | DETAILS |
|---------------|---------------------------------|--|
| Enable | <code>/etc/nsswitch.conf</code> | Vanilla: <code>hosts: files dns</code> Module: <code>hosts: files getdns</code> |
| Global Config | <code>/etc/getdns.conf</code> | Default configuration is: <code>dnssec: validate</code> (Also available: <code>dnssec: secure only</code>) |

Possible to deploy and configure via automated tools (ansible/puppet)

POC Per-User Configuration

- Evaluate Whether Appropriate to Target Users



The screenshot shows a web browser window with the address bar displaying "getdns-config.localhost". The page title is "GETDNS local settings". The interface is divided into three main sections: "DNSSEC settings", "TLS settings", and "Logging". Each section contains radio button options for configuration. At the bottom, there is a red warning message and a "Save Preferences" button.

GETDNS local settings

DNSSEC settings

- Accept Insecure, but reject bogus answers
- Accept ONLY SECURE answers

TLS settings

- Fail if no TLS
- Try TLS with fallback to TCP/UDP
- No TLS

Logging

- Critical errors
- Warnings
- Info (start, changes)
- Verbose

Browsers will need to be restarted for changes to apply.

[Save Preferences](#)

Signalling

The standard library calls such as `getaddrinfo()` have an existing interface and error messages.

DNSSEC ISSUE: In the case of an invalid signature on a DNS record a generic error must be returned (SERVFAIL).

There is no way of telling the application that an answer was received, but it had an invalid signature!

It would be beneficial to be able to signal to the user that an answer was received, but was rejected as it was insecure.

Signalling (continued)

In this POC we experimented with approaches for signalling

e.g. For a simple HTTP page the browser could be redirected to a page server locally that informs the user of the error.

While this approach is limited to simple pages server on port 80 it serves to prove the benefit of signalling

Other approaches such as browser plugins and system tray notifications that can communicate with the NSS library are being investigated.

Future Work

- Improved signalling
- Addition of system cache to improve performance
- DNS-over-TLS
 - Configurable including policies for fall backs:
tls: prefer_tls / require_tls / disable_tls
- More fine grained configuration policy
- Research into advanced security eg. SELinux, AppArmour & containers
- Windows and MAC OS X module architecture

Upstream Engagement

- Work with open source community to deliver open source stub resolver with ***getdns*** and NSS module
- Work with Linux, BSD, Java and open source app server communities to embed resolver so it's available “out of the box”

Summary

- Solution enables DNSSEC validation for legacy applications via existing UNIX & Linux name resolution framework
- Integrated in the Operating System using the nsswitch interface available on various flavors of Linux and BSD and many other UNIX-like platforms.
- The design uses the ***getdns*** library to handle all the DNS related functions (getaddrinfo(), etc).

DEMO AVAILABLE ON REQUEST!

Thank You

Questions?

Please forward questions and comments to:

stub-resolver AT verisignlabs DOT com