Client side DNSSEC validation

Red Hat
Tomáš Hozza
thozza@redhat.com
2014-05-13

Agenda

1 Motivation

2 Requirements

3 Solution Architecture

4 Conclusion

Section 1

Motivation

Motivation

- Trusted data in DNS (TLSA, SSHFP, IPSECKEY)
- Attacks on plain DNS
- Authenticated DNS data for applications
- End-to-end DNSSEC validation

Section 2

Requirements

Local validating resolver

- Requirement for trusted (local) validating resolver
- Resolver has to support functionality required by other requirements

Resolver reconfiguration mechanism

- Respond to dynamic network changes
- Communicate with the system network connection management system

Split DNS configuration

- For networks with multiple DNS views
- Usually needed for VPN connections
- Provided nameservers may not fully support DNSSEC - What then?

Network-provided nameservers probing

- Testing functionality of DHCP/VPN provided nameservers
- Should support
  - UDP/TCP query replies
  - EDNS
  - AD, DO bits
  - RRSIG, DS, DNSKEY, NSEC/NSEC3 records
- User decides what to do

Fall-back configuration

- In case network-provided nameservers don’t support DNSSEC properly
- Bypass network port filtering (e.g. using ports 80/443)

Captive portal detection

- DNSSEC would cause issues
- Detect such situation
- Proper handling

Section 3

Solution Architecture

Solution Architecture

- Validating resolver – unbound
- Reconfiguration mechanism – dnssec-trigger
- Network connections manager – NetworkManager

Current situation

NetworkManager

- On every network change runs dispatcher scripts
- Provides API for reading network configuration

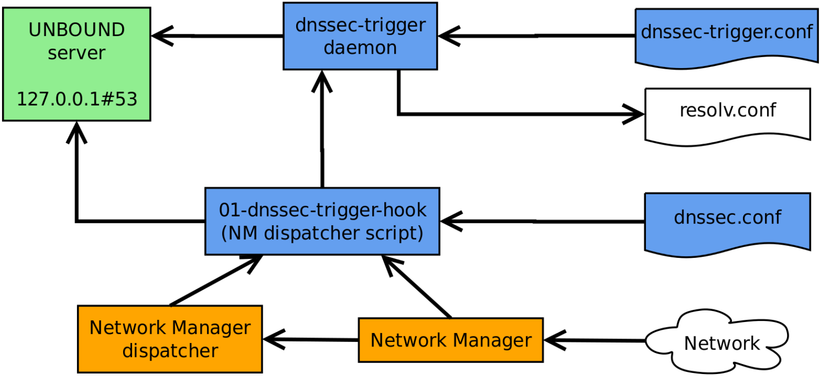
dnssec-trigger

- Provides NM dispatcher script
- Handles – nameservers probing, captive portal detection, fall-back configuration, split DNS
- Rewrites resolv.conf

unbound

- Reconfigured by dnssec-trigger (global forwarders)
- Reconfigured by dnssec-trigger dispatcher script (forward zones)

Current situation



Future plans

NetworkManager

- Use configuration used by previous solution
- Provide better and extended configuration possibilities
- Rewrite resolv.conf

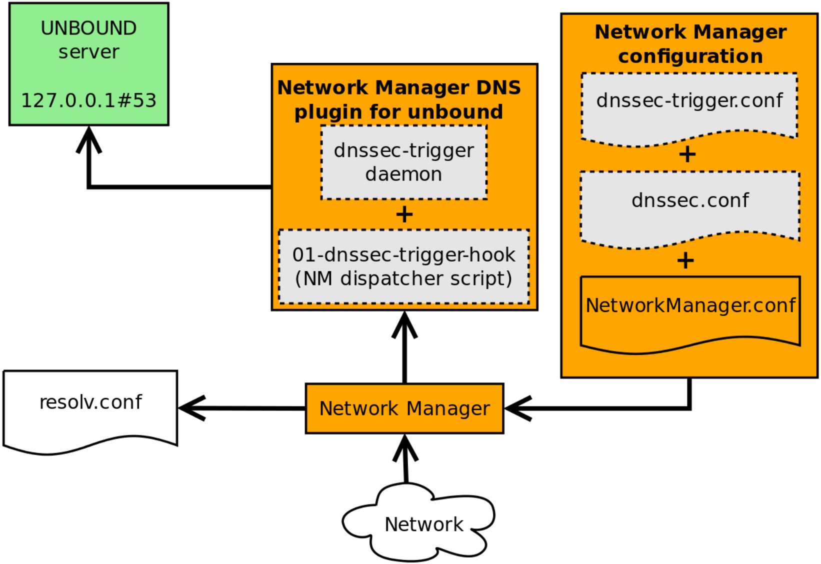
unbound NetworkManager DNS plugin

- Incorporate dnssec-trigger's (and dispatcher script) functionality
  - nameservers probing
  - captive portal detection and handling
  - split DNS configuration
  - reconfigure unbound

unbound

- Reconfigured by unbound NetworkManager DNS plugin

Future plans



UNBOUND server

127.0.0.1#53

Network Manager DNS plugin for unbound

dnssec-trigger daemon

+

01-dnssec-trigger-hook (NM dispatcher script)

Network Manager configuration

dnssec-trigger.conf

+

dnssec.conf

+

NetworkManager.conf

resolv.conf

Network Manager

Network

Section 4
Conclusion

Conclusion

- End-to-end DNSSEC validation is important for client side applications using trusted DNS data
- Clients and workstations work in dynamic environment and need special approach
- Described requirements on the client side DNSSEC validation solution
- Described solution used in Fedora project
    - Present – unbound + dnssec-trigger + NetworkManager
    - Future – unbound + unbound NM DNS plugin + NetworkManager

thozza@redhat.com