Monitoring DNSSEC

Martin Leucht <martin.leucht@os3.nl>
Julien Nyczak <julien.nyczak@os3.nl>
Supervisor: Rick van Rein

System and Network Engineering 2015
Introduction

- DNSSEC becomes more and more popular
- Expired RRSIG RR might result that zone not available
- Need for monitoring
- Monitoring systems exist but are too specific to be widely deployed
- Solution: Monitoring DNSSEC through SNMP
SNMP

- standard application protocol to manage and monitor devices running on IP network
- can be implemented for applications as well
- agent-manager architecture
- structure of the management information and SNMP variables defined in a Management Information Base (MIB)
- SNMP variables are assigned to Object Identifiers (OID) in a hierarchical manner
Research Questions

- What are vital life signs for monitoring DNSSEC?
- How to construct a MIB module for DNSSEC?
- How to conduct monitoring based on such a MIB?
- How do architectures for monitoring DNSSEC compare?
Approach (1/2)

- To be independent on other software components (only AXFR and authoritative queries)

- Vital life signs for DNSSEC
  - Availability of a zone from a resolver point of view (initial check)
  - Verify DNSKEY RRSIG against published KSK
  - DS record count = delegation count (in a parent zone)
  - TTL checks
  - List of name servers for a zone
  - Expiration date of RRSIG for SOA, NS, DNSKEY
  - Discrepancies in serial numbers between slave and master (slave serving expired data)
Approach (2/2)

- Construct the MIB based on vital life signs
- Write the SNMP subagent (python-netsnmpagent)
- How data is retrieved from zones?
  - From a central repository: XML
  - DNSSEC data collected via AXFR requests, DNS queries to authorities and resolvers
OID entry point inside ARPA2 OID tree (enterprise OID 44469):
- ARPA2-Experimental-DNSSEC-MIBv1
- .1.3.6.1.4.1.44469.666.53.46.161.1
Objects are defined using a subset of Abstract Syntax Notation One (ASN.1) called "Structure of Management Information Version 2 (SMIv2)" RFC 2578

Objects organized in columnar (conceptual tables) or scalar objects.

Four tables indexed by domain name (OCTET-STRING)
- dnssecZoneGlobalTable, dnssecZoneAuthNSTable, dnssecZoneSigTable, dnssecZoneDiffTable

Datatype INTEGER to represent boolean and numeric values, OCTET-STRING to represent strings (e.g. domain names)

Usage of Textual conventions to customize object-types
DNSSEC MIB implementation (3/4)
ARPA2-Experimental-DNSSEC-MIBv1::dnssecZoneGlobalServFail."derby.practicum.os3.nl" = INTEGER: noerror(1)
DNSSEC MIB implementation (4/4)

- `dnssecZoneGlobalIndex` OBJECT-TYPE
  - SYNTAX DomainOctetString
  - MAX-ACCESS not-accessible
  - STATUS current
  - DESCRIPTION "Reference index for each observed zone"
  
- `dnssecZoneGlobalServFail` OBJECT-TYPE
  - SYNTAX CustomInteger
  - MAX-ACCESS read-only
  - STATUS current
  - DESCRIPTION "Indicates that ..."

- `DomainOctetString` ::= TEXTUAL-CONVENTION
  - DISPLAY-HINT "255t"
  - STATUS current
  - DESCRIPTION "An octet string containing characters in UTF-8 encoding."
  - SYNTAX OCTET STRING (SIZE (1..255))

- `CustomInteger` ::= TEXTUAL-CONVENTION
  - STATUS current
  - DESCRIPTION "Convention for return values of Integer variables."
  - SYNTAX INTEGER { noerror(1), error(2), unknown(3) }

- `ARPA2-Experimental-DNSSEC-MIBv1::dnssecZoneGlobalServFail."derby.practicum.os3.nl"` = INTEGER: noerror(1)

- `22 = number of characters`

- `ASCII values (decimal) for "derby.practicum.os3.nl"`
NET-SNMP toolkit → de-facto standard for SNMP implementations on most OS

- Includes applications (snmpget, snmpwalk, etc.) and libraries
- Includes C API to write own AgentX subagents [RFC 2741](https://tools.ietf.org/html/rfc2741)
- Subagents register to snmpd master agent via Unix socket
AgentX SNMP subagent based on Python NET-SNMP API module “netsnmpagent” written by Pieter Hollants licensed under GPLv3

**Warning:** Consider this when using our prototype!

- UpdateSNMPObj() function is self written
- Subagent is capable of most SNMP data types
- Handles requests for our DNSSEC MIB
- Allows to register, update and clear table rows and scalar values
- Subagent works asynchronously, data update thread is decoupled from data providing thread
- Data for subagent is provided by two main wrapper scripts (dnspython)
SNMP subagent implementation (3/4)
SNMP subagent implementation (4/4)

- **NET-SNMP Master AgentX**
- **Python AgentX subagent**
- **DNSSEC-MIB**

Diagram:
- Server 1: `ns1.warsaw.practicum.os3.nl`
  - 145.100.104.62/27
  - AXFR zone transfers

- Server 2: `derby.practicum.os3.nl`
  - 145.100.104.165/27
  - AXFR

**DNSSEC signed zones**
- Master for: `warsaw.practicum.os3.nl`
- Master for: `paris.derby.practicum.os3.nl` (delegation)
- Slave for: `derby.practicum.os3.nl`
- Point of Delegation: `berlin.warsaw.practicum.os3.nl`

**DNSSEC signed zones** (right)
- Master for: `derby.practicum.os3.nl`
- Master for: `berlin.warsaw.practicum.os3.nl` (delegation)
- Slave for: `warsaw.practicum.os3.nl`
- Point of Delegation: `paris.derby.practicum.os3.nl`
Conclusion / Future Work

- Proof of concept based on SNMP to cover critical data of DNSSEC signed zones
- Conduct monitoring based on proof of concept
- SNMP Notifications/Traps
- Expand MIB to cover more DNSSEC related data
  - Validation of all RRSIG RR (expired/non validated)
  - Check for broken NSEC3 chain
  - …
Demo