DANE verification test suite

Hamza Boulakhrif Guido Kroon

Supervisor:
Michiel Leenaars (NLnet Foundation)

hamza.boulakhrif@os3.nl, guido.kroon@os3.nl

University of Amsterdam

Faculty of Physics, Mathematics and Informatics
Graduate School of Informatics
System and Network Engineering MSc

February 6, 2015

Introduction

- Classic CA model
  - Trusted Certificate Authorities
  - Pre-configured CA certificate collections
- DANE
  - DNSSEC chain of trust
  - TLSA RRs
  - PKIX validation (optional)

Classic CA model

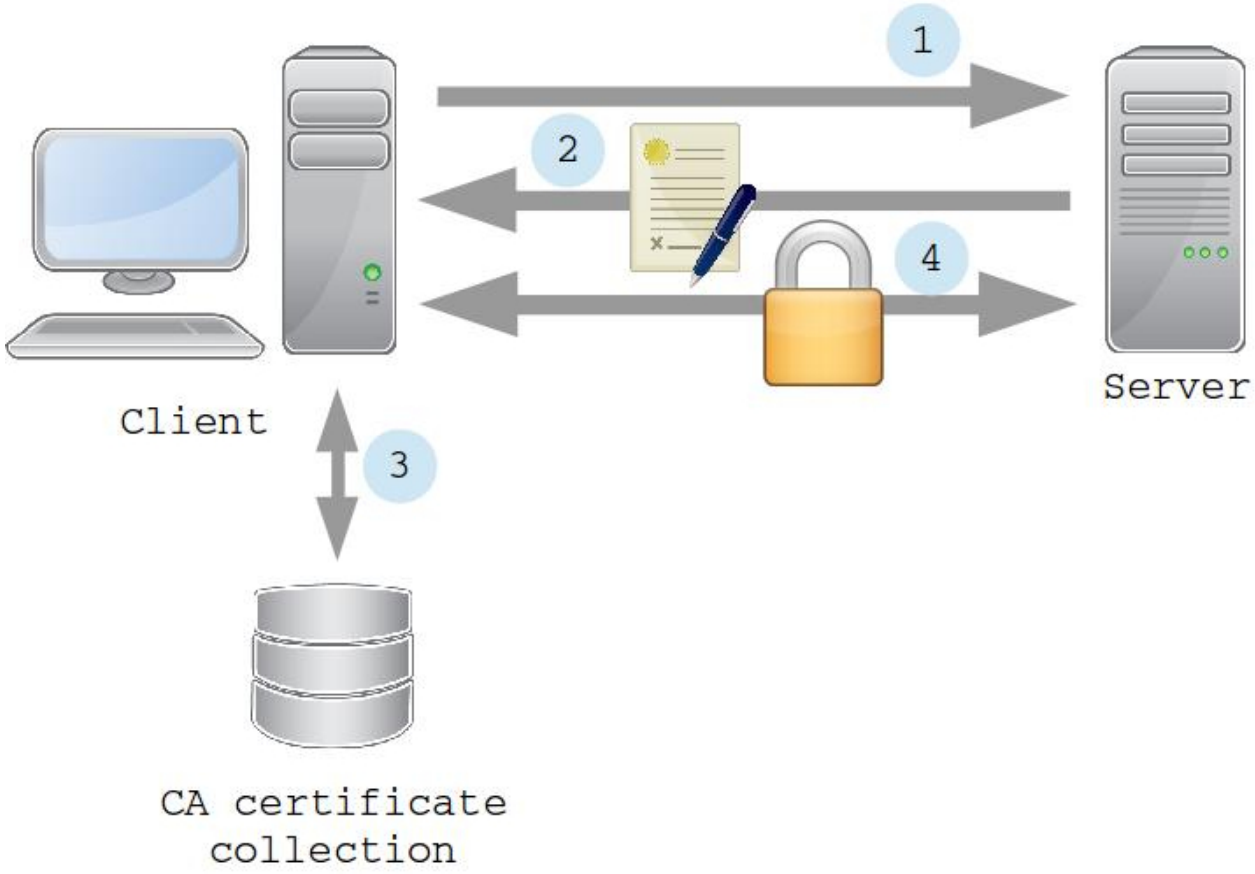


Figure 1: Classic validation.

DANE model

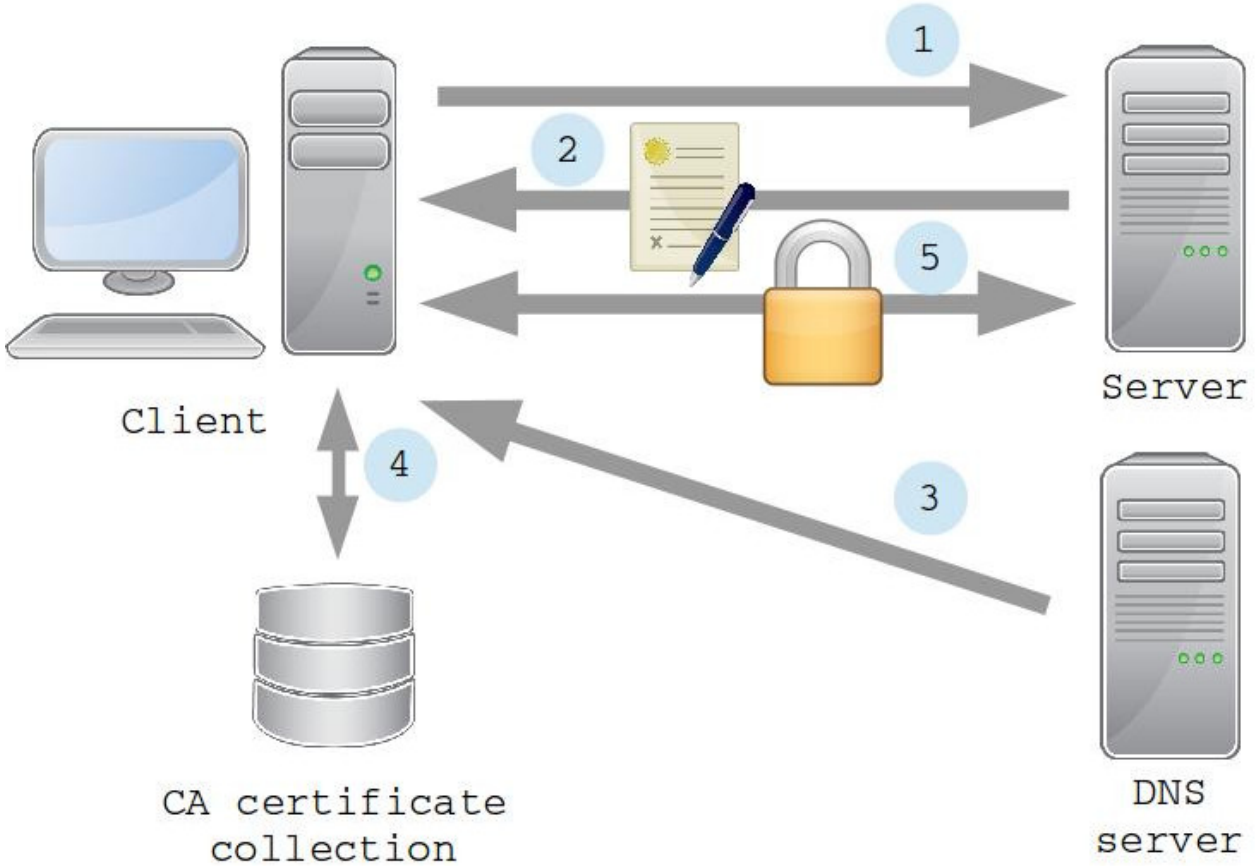


Figure 2: DANE validation.

TLSA RR

- Basically a customised SRV RR
  - Service, Proto, Name, Class fields
  - Certificate Usage
  - Selector
  - Matching Type
  - Certificate Association Data

TLSA RR format

```
_Service._Proto.Name Class TLSA Usage Selector Mtype Data
```

TLSA RR example

```
_443._tcp.dane.internet.nl. IN TLSA (
    0 0 1 d2abde240d7cd3ee6b4b28c54df034b9
          7983a1d16e8a410e4561cb106618e971 )
```

Certificate Usages (1)

The four different Certificate Usages of DANE.

- **Usage 1 (Server Certificate Constraint)**
  TLSA RR specifies which EE certificate should be used for the domain.
- **Usage 3 (Domain-issued Certificate)**
  TLSA RR specifies the TLS certificate that should be used for the domain, without PKIX validation.

Certificate Usages (1)

- **Usage 0 (CA Constraint)**
  TLSA RR specifies which CA will provide TLS certificates for the domain.
- **Usage 2 (Trust Anchor Assertion)**
  TLSA RR specifies which trust anchor will provide TLS certificates for the domain, allowing the use of a CA not included in the CA certificate collection of the application.

Research question

Can a test suite be devised to allow developers and implementers to validate the reliability and consistency of an implementation of DANE, and its ability to correctly handle unforeseen input or deviations from the official TLSA syntax as per RFC 6698?

Scope

The scope for this research.

- Analysis of RFC6698
- Extensible test suite
  - Usages
- Test DANE implementations

Not part of scope research:

- (Re)writing DANE-tools
- (Re)compiling of DANE-tools

Approach

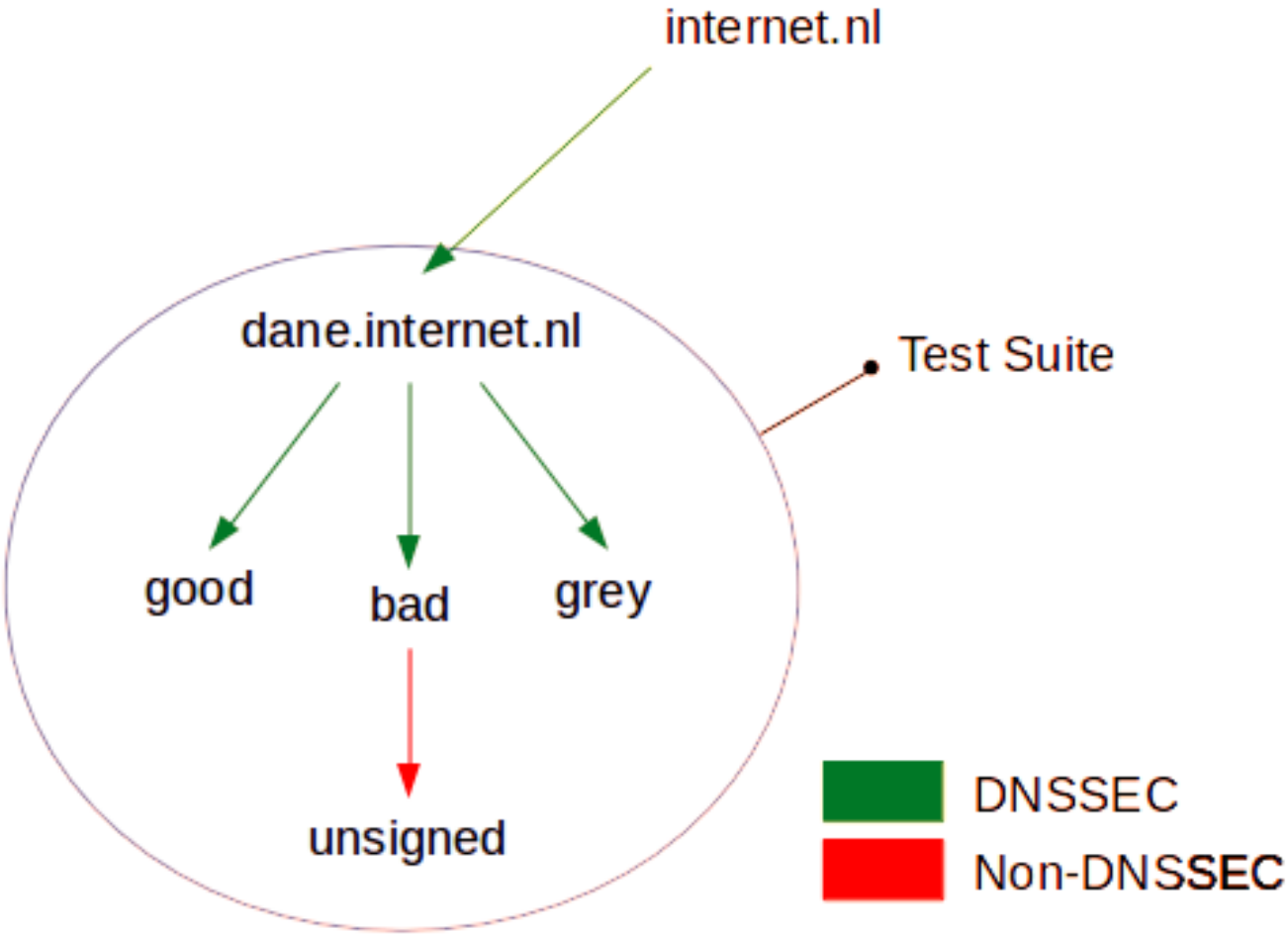
The approach for this research.

- Analysis of DANE RFC 6698 (and RFC 6394)
- Deployment of environment
- Build test suite in environment
- Test DANE implementations

Test suite

The test suite is built by using:

- BIND
- Apache



Experiments (1)

- GnuTLS
- ldns-dane
- DNSSEC/TLSA Validator (browser add-on)

```
[redacted]@desktop-24:~$ danetool --check falsecacert.bad.dane.internet.nl
Resolving 'falsecacert.bad.dane.internet.nl'...
Obtaining certificate from '2a04:b900:0:100::29:443'...
Querying DNS for falsecacert.bad.dane.internet.nl (tcp:443)...
_443._tcp.falsecacert.bad.dane.internet.nl. IN TLSA ( 01 00 01 ef2bc46a93cc5f17a
054ac9a06e0b1b98061896f0f288d1826e8634834e3d1ca )
Certificate usage: End-entity (01)
Certificate type:  X.509 (00)
Contents:          SHA2-256 hash (01)
Data:              ef2bc46a93cc5f17a054ac9a06e0b1b98061896f0f288d1826e8634834e3d
1ca

Verification: Certificate matches.
[redacted]@desktop-24:~$
```

Figure 4: GNUTLS Danetool

Experiments (2)

Test cases that are devised by the analysis of the DANE specification.

- (Non-)existing usages
- (Non-)existing Selectors
- (Non-)existing Matching types
- Combination of Selector and Matching type incorrect
- (In)correct hash (type)
- Expired certificates
- Unsigned DNSSEC chain
- Wildcard usage
- Incorrect signed certificates

Results

- GnuTLS
  - No PKIX validation (intentional).
- ldns-dane
  - Specify CA certificates manually for PKIX validation.
- DNSSEC/TLSA Validator
  - No PKIX validation, even though it claims to.

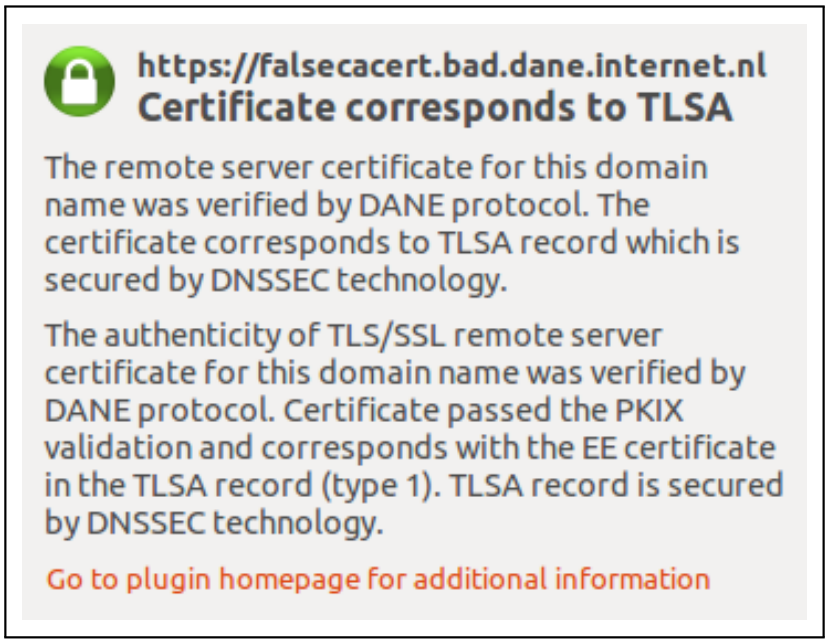


Figure 5: DNSSEC/TLSA Validator without proper PKIX validation.

Conclusion

Based on the results, a couple of conclusions can be derived.

- RFC 6698
  - Interpretation
- Test suite
  - Good
  - Bad
  - Grey
- BIND
  - Test cases
  - Limitations

Future work

Some noteworthy details, which lie outside of the scope of this project:

- Think of more test cases
  - Proxy in front of BIND
- Test cases for all usages (CA Contraint)
- Source code analysis of DANE implementations
- Complete DANE support in DANE implementations

The End