UNIVERSITY OF AMSTERDAM
SYSTEM AND NETWORK ENGINEERING

Research Project 1:

Implementing a DANE validator

Author:
Pieter Lexis
pieter.lexis@os3.nl

Supervisor:
Bert Hubert
bert.hubert@netherlabs.nl

Abstract

In this paper, I look at the implementability of the 14th draft of the DNS-based Authentication for Named Entities (DANE) specification. To this end a tool has been made and released (called `swede`) to create and verify `TLSA` records. All permutations of `TLSA` records were put into DNS and successfully verified end-to-end using a TLS service. Apart from a current discussion within the DANE Working Group about the definition of "pass PKIX validation" and how it relates to usage 2, DANE is implementable.

February 9, 2012

Acknowledgments

I would like to thank the following for their contributions:

- Bert Hubert - For his insight into DNS, and supervising
- The DANE working group at the IETF - For answering my questions
- The people in `#powerdns` on OFTC - For their DNS knowledge and support
- The proofreaders

Introduction

The current system used for trust on the Internet depends on certificates issued by Certificate Authorities. In this system there is no way of allowing only a specific CA to sign certificates for an organization or service. Unfortunately, end-user applications trust a large number of these CAs. This makes CAs targets for crackers wanting to impersonate a secured service. Several projects have been started to combat this problem; the main problem seems to be the secure, scalable distribution of certificates and their metadata.

Although Domain Name System Extensions (DNSSEC) are currently being deployed by the Top-Level-Domain (TLD) operators around the world, a 'killer' application for this technology does not exist yet. However, the guarantee that data received from the DNS has not been tampered with opens the way for adding more than address records into DNS. The inclusion of cryptographic information for applications is one of them.

The DNS-based Authentication of Named Entities (DANE) working group of the IETF[1] has been founded to create standards to accomplish this. As per the charter:

> Specify mechanisms and techniques that allow Internet applications to establish cryptographically secured communications by using information distributed through DNSSEC for discovering and authenticating public keys which are associated with a service located at a domain name.

Motivation

As shown in the coming chapter, the current trust system used on the internet suffers from drawbacks. The DANE specification allows the creation of an out-of-band system to pin certificates to DNS names. Because the specification is still in development is makes for an interesting research topic.

In a previous OS3 Research Project, Danny Groenewegen and Pieter Lange implemented a DNSSEC validator with DANE support in a Firefox plugin [6]. This proved that it is not hard to implement this validation in the end-user browser. This project will look at the other end of DANE, namely the deployment side.

[1] Internet Engineering Task Force, the organization creating Internet standards

Chapter 1

Trust on the Internet

The current technology used for trust management and secure connections on the Internet is mostly based on the Secure Sockets Layer (SSL) and Transport Layer Security (TLS) communication standards, collectively called TLS in this report, combined with the use of X.509 certificates. In the last few years, problems with this infrastructure have come to the surface. The IETF is drafting specifications to combat one of these problems.

The problem with the current trust infrastructure

The trust infrastructure on the Internet is based on the Public Key Infrastructure using X.509 version 3 certificates (PKIX) and is defined in RFC5280 [1]. This infrastructure relies on Certificate Authorities (CAs) to certify that a cryptographic key-pair belongs to a certain entity (in this report this entity is limited to 'end-machine hostname').

When a program initiates a TLS connection, it is presented with a PKIX certificate for that service. The user's program verifies that the certificate is issued by a CA it trusts and has the name in the Subject field of the certificate matches the name of the server it connects to. User's browsers and other applications that are able to use TLS come with a large number of built-in trusted CA certificates [3, p. 19].

A CA can issue certificates for any domain name. This means that a certificate for `mybank.example.com` signed by a CA is equally valid as one signed by another CA. If an attacker could get a CA to issue a certificate stating an attacker-controlled key is for `mybank.example.com`, this attacker could use DNS Cache-poisoning[9] to send users to a seemingly valid, but fake bank website. This property makes CAs targets for crackers wanting to obtain false certificates for important domains (like *.google.com or update.microsoft.com). Lately, this has happened with DigiNotar[1] and Comodo[2].

The need for a solution

A solution to this problem is 'pinning'[4] a public key or certificate out of band, to make sure only certain CAs can issue certificates for certain (sub-)domains or organizations.

The HTTP Strict Transport Security (HSTS) [12] specification [7] provides a way to mandate the use of a secure connection. Browser vendors complement this specification by supplying whitelists of which CA may sign certificates for certain (sub-)domains [3]. This is, however, not a scalable solution.

Another proposed solution is the 'sovereign keys' project by the Electronic Frontier Foundation [5] (EFF). This solution that uses a "semi-centralized, verifiably append-only data structure" containing the keys and revocations. These keys can only be added when it is strongly verified that the domain belongs to the requesting party. A browser would, when connecting to an TLS service, lookup the certificate from this key-store.

Another way of solving the problem is using 'multi-path probing'[13] to ensure the correct certificate is offered to the end-user. When a user contacts a TLS service, it sends a request to a number of trusted 'notaries', these notaries also connect to that service and send (the hash) of the certificate to the end-user. The user can then validate if it

[1] `http://arstechnica.com/security/news/2011/08/earlier-this-year-an-iranian.ars`
[2] `http://www.infoworld.com/t/authentication/weaknesses-in-ssl-certification-exposed-comodo-security-breach-593`
[3] `http://dev.chromium.org/sts`

is connecting to the right service. The first project to implement this behavior is the Perspectives Project [11] from the Carnegie Mellon University. Recently, security researcher Moxie Marlinspike created a browser plugin called Convergence [10] to implement this behavior in web-browser while maintaining end-user privacy from the notaries.

Yet another possible solution is adding certificate information to DNS, leveraging the existing DNSSEC trust to authenticate this data. This solution is called DNS-based Authentication of Named Entities (DANE).

The DANE working-group of the IETF is currently drafting this specification and it appears to be nearing completion. This report is focused on this specification and will not discuss other solutions.

Chapter 2

DANE

Note: this section describes the 14th draft of the DANE specification, which may have been superseded. Please consult the latest version for implementation details. See the Addendum for updates.

2.1 Use cases

Before starting work on the specification, the working group has set forth a document containing use-cases and demands for this specification [2]. This document describes 3 major use cases [2, Section 3] and their associated certificate constraints:

2.1.1 CA Constraints

This constraint should limit the number of CAs that can issue certificates for an organization. It should allow the service maintainer to express "The certificates for my services must be signed by MyTrustedCA".

2.1.2 Service Certificate Constraints

When a CA should secretly issue new certificates for the service, there should be a way a maintainer can express "This specific certificate is the only valid one for this service" to the end-user.

2.1.3 Trust Anchor Assertion and Domain-Issued Certificates

The last use case deals with the ability to create private CAs and allow the use of self-signed certificates that will be considered valid by the application initiating a TLS connection.

2.2 DANE specification

The DANE specification [8] describes a new DNS record that contains 'certificate association data'. A compliant implementation will, before sending any information to the service, look up this DNS record and validate the certificate from the TLS service against it. If there is a match, and the constraints set in the record are met, the connection is resumed. Otherwise it is aborted without the ability for the user to 'click-through' the warning.

DANE does not mandate the use of DNSSEC when deploying it, it does recommend it as it leverages the trust provided by DNSSEC as an out-of-band (i.e. not within the PKIX infrastructure), authenticated mechanism to distribute certificate association data.

The new DNS resource record is called `TLSA` (for 'TLS Association'). The name belonging to the record contains the port and the protocol on which the TLS service resides. A truncated example is shown below:

```
_443._tcp.dane.kiev.practicum.os3.nl IN TLSA ( 1 0 1 E8EB600F620...)
```

2.3 TLSA record data

A TLSA record consists of 4 fields, the first three describing the fourth (the actual association data):

- 1 octet 'usage' field
- 1 octet 'selector' field
- 1 octet 'matching type' field
- A variable length 'certificate for association' field

A visual overview of the wire-format is shown in 2.1.

```
                     1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Usage     |    Selector   | Matching Type |               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+               /
/                                                               /
/                   Certificate for Association                 /
/                                                               /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2.1: A TLSA record (wire-format)

2.3.1 Usage

The usage field describes how the association data should be treated, this is known as a 'constraint'. There are 3 values defined for this field:

- 0 - CA constraint
- 1 - End Entity constraint
- 2 - 'Domain-Issued' certificate

Usage 0 and 1 mandate that the certificate presented chains to a valid CA certificate. So apart from matching the TLSA record, the certificate must be issued by a trusted CA.

With usage 0, the certificate in the TLSA record must be a CA certificate that is trusted by the end-user's program. This allows for only one CA to sign certificate for a service. It also means that the services' certificate can be renewed without changing the TLSA record.

With usage 1, the certificate in the TLSA record must match the one received from the TLS service and it must be issued by a trusted CA.

Usage 2 is somewhat unclear in this revision of the specification (see section 4.5 for the issues). The certificate in the the record is either a CA or an End Entity certificate and it must be used as a trust-anchor when constructing the certificate chain.

2.3.2 Selector

The selector indicates *what* should be matched, it is either

- 0 - The full certificate
- 1 - The SubjectPublicKeyInfo of the certificate

When selector 1 is used together with usage 1, an administrator can change CAs without updating its `TLSA` records if he uses the same key to create a new request.

When selector 1 used in combination with usage 0, the CA can issue a new certificate for itself (using a stronger hashing algorithm or another name) without the domain's administrator having to change the `TLSA` record.

There are security considerations with the use of selector 1, these are documented in Appendix A.1.2 of the specification [8].

2.3.3 Matching Type

The matching type indicates how the association data should be matched to the certificate from the TLS service:

- 0 - Byte-by-byte comparison
- 1 - SHA-256 match of the certificate
- 2 - SHA-512 match of the certificate

2.3.4 Certificate for Association

This contains the bytes to-be matched, the name is slightly incorrect and will be known as 'Certificate Association Data' in the next draft as it can contain a hash or SubjectPublicKeyInfo instead of a full certificate.

Chapter 3

Research

The research is split into two parts, the first part was focused on implementing the DANE specification into a tool that can create and verify `TLSA` records and verify the validity of them by checking the record against the certificate(-chain) offered by the TLS service. The second part was focused on testing this validation in a real-world, end-to-end situation: on an HTTP server offering TLS services.

3.1 Research Question

The question that this project tried to answer is "Is DANE in its current form implementable and does it achieve its goal of securely binding DNS names to TLS certificates on end-hosts?". It was expected that this is the case.

A side-effect of the research and in order to help the specification move forward, the results might be used as test vectors for the specification. And the server used for testing could remain operational for a time as a test bed to help implementers test their DANE implementations.

3.2 Tooling

Tooling had to be created to create and verify `TLSA` records. The only tool available was the `dane` script from the `sshfp` [14] package. Unfortunately, this only supported an older draft of the DANE specification *and* only could create 1 type of record (usage 1, selector 0, matching type 1). The tooling required needed at least the following features:

- Create all 18 permutations of `TLSA` records, with the ability to:
  - load certificates from disk and from the TLS service
  - create draft (`TYPE65468`) and RFC (`TLSA`) records
- Verify a `TLSA` record
  - Securely receive the `TLSA` record and address record from DNS (with the option to do this insecurely)
  - Compare the record with the certificate retrieved from the TLS session

By writing the tooling, it was possible to check if the specification was well-written and contains no ambiguities or oversights. This became an important part of the research.

3.3 End-to-end testing

In order to test whether DANE could be deployed in the real world, all 18 permutations of record would have to be created, added to DNS and verified using TLS connections. This needed 2 services, a nameserver and a TLS enabled server (a webserver).

Table 3.1: Values for the fields in the TLSA record and the certificate used for every port

Port number	Usage	Selector	Matching Type	Certificate used	Other information
1500	2	0	0	Self-signed	
1501	2	0	1	Self-signed	
1502	2	0	2	Self-signed	
1503	2	1	0	Self-signed	
1504	2	1	1	Self-signed	
1505	2	1	2	Self-signed	
1506	1	0	0	Comodo PositiveSSL	
1507	1	0	1	Comodo PositiveSSL	
1508	1	0	2	Comodo PositiveSSL	
1509	1	1	0	Comodo PositiveSSL	
1510	1	1	1	Comodo PositiveSSL	
1511	1	1	2	Comodo PositiveSSL	
1512	0	0	0	Comodo PositiveSSL	
1513	0	0	1	Comodo PositiveSSL	
1514	0	0	2	Comodo PositiveSSL	
1515	0	1	0	Comodo PositiveSSL	
1516	0	1	1	Comodo PositiveSSL	
1517	0	1	2	Comodo PositiveSSL	
1518(1)	0	0	2	Comodo PositiveSSL	Two valid TLSA records for the same hostname
1518(2)	1	0	2	Comodo PositiveSSL	idem
1519	2	0	1	Self-signed	CNAME the hostname[a]
1520	2	0	1	Self-signed	idem[b] *and* CNAME the TLSA record name[c]
1521	9	6	3	None	Deliberately invalid TLSA record

[a]cname1.dane.kiev.practicum.os3.nl → dane.kiev.practicum.os3.nl
[b]cname2.dane.kiev.practicum.os3.nl → dane.kiev.practicum.os3.nl
[c]_1520._tcp.cname2.dane.kiev.practicum.os3.nl → _1520._tcp.dane.kiev.practicum.os3.nl

3.3.1 Setup

All experiments were performed on Debian 6 (“Squeeze”) using, where available, packages from the Debian repositories.

To serve the created TLSA records, a patched PowerDNS 3.1-pre[1] was used. In order to offer the certificates inside an SSL session, the Apache Webserver was used.

3.3.2 Method

After creating the TLSA records, putting them into DNS and signing the zone, a total of 21 TCP ports were used to offer an SSL service. There were 2 certificates used, the first one was self-signed and the other one was signed by Comodo PositiveSSL. For an overview of the used ports and their TLSA field values, see Table 3.1.

The self-signed certificate was offered on the ports used for all usage 2 records, whereas the Comodo signed certificate and the corresponding certificate chain was offered for usage 0 and 1.

The TLSA records for the first 18 ports were all permutations of the usage, selector and matching type fields. One port was used to offer both a usage 0 and a usage 1 TLSA record. Two ports were used to test the tool’s ability to handle CNAME redirection, as described in paragraph A.2.1.1 of the DANE specification[8].

A final record was added with illegal values of the fields, to check the record validation code in the tool.

After setting up the DNS and webserver, all records were verified using the tool. As all records were valid, all records with their subsequent retrieved certificates should verify successful. The DNS entries used during the experiment can be found in Appendix A.

[1]http://www.powerdns.com

Chapter 4

Results

This chapter describes the results of the tests performed and the issues that arose during the tests or the implementation of DANE.

4.1 `swede`

The first outcome of this research is `swede`, a tool that can create and verify `TLSA` records. It has the features listed in section 3.2. It is free software and can be obtained from `https://github.com/pieterlexis/swede`.

Swede is written in python and uses the python-bindings for libunbound[1] for secure lookups, M2Crypto.SSL and M2Crypto.X509[2] for SSL and certificate related work and IPAddr[3] for verification of A and AAAA records.

Swede will be developed further, supporting newer drafts of the DANE specification when they are published. Paul Wouters[4] wants to include `swede` in a new package named '`secdns`' that will contain tools to create DNS records that include certificates or other cryptographic information.

4.2 Real-world tests

During the real-world tests all 18 permutations of records validated, as shown in Appendix B.2. The verification of the other records can be found there as well.

To verify that `swede` also works on records not created by itself, several verifications were done on hostnames posted to the DANE mailing list[5] before the release of `swede`, these verifications were also successful as seen in Appendix B.3.

This shows that DNS based association of certificates to services is becoming a viable option to solve a part of the issues with trust management on the internet.

4.3 Patches for PowerDNS

During the course of this research, PowerDNS 3.1-pre was selected as the nameserver to use because of its preliminary support for `TLSA` records. This support was however incomplete, as it treated the certificate for association field data as base64 (in accordance with an older draft) instead of hexadecimal. This was fixed in two separate commits[6,7].

[1]`http://unbound.net/documentation/pyunbound/index.html`
[2]`http://chandlerproject.org/bin/view/Projects/MeTooCrypto`
[3]`https://code.google.com/p/ipaddr-py/`
[4]maintainer of the `sshfp` package and creator of first `TLSA` creation tool
[5]`http://www.ietf.org/mail-archive/web/dane/current/msg04114.html`
[6]`http://wiki.powerdns.com/trac/changeset/2347`
[7]`http://wiki.powerdns.com/trac/changeset/2358`

4.4 Test bed and test vectors

The server at `dane.kiev.practicum.os3.nl` will remain online as a test bed. There are `TLSA` records served and ports opened for TCP ports 1500 through 1521. See table 3.1 for the records.

The created records and certificates used will be offered to the DANE working group for inclusion in the specification as test vectors.

4.5 Implementation issues

During the course of this research only one real issue has come to the surface, this issue stems from the term "pass PKIX validation" in combination with usage 2 and is still being discussed within the DANE working group.

4.5.1 Usage 2

In paragraph 2.1.1 of the DANE specification, usage 2 is defined as

> "The target certificate MUST pass PKIX validation, with any certificate matching the TLSA record considered to be a trust anchor for this validation"

Later on, in paragraph 4.3 it is explained as

> "Certificate usage 2 is used to specify a certificate, or the public key of such a certificate, that must be used as a trust anchor when validating the end entity certificate given by the server in TLS. This usage is sometimes referred to as "domain-issued certificate" because it allows for a domain name administrator to issue certificates for a domain without involving a third-party CA"

This appears to mean 'Create a record form a certificate in the chain (even of length 1 [8]) and bypass regular certificate validation', this is however never mentioned explicitly. This could make usage 2 an under-used option in the specification, where it has the potential to give complete control to the domain-owners for issuing certificates. When asking what exactly usage 2 is for, the answer was "[..] usage 2 lets you specify an end-entity certificate that is used as a trust anchor" [9]

4.5.2 PKIX validation

The term 'PKIX validation' is not clearly defined by the IETF or the X.509 specification. According to a post on the mailing list[10], it means "The certificate chain must be traversed successfully". In other emails, it is said to mean that a program must implement the algorithm described in section 6 of RFC5280 [1].

4.5.3 `swede`'s interpretation

While creating `swede`, I used the following definition of usage 2 and PKIX:

> "Any certificate in the valid certificate-chain offered by the SSL/TLS service MUST match the `TLSA` record."

There is discussion in the working group about the definition, when there is consensus `swede` will be updated to reflect this new definition.

Apart from a better definition of usage 2, the working group is planning to include a new usage (3) to the next draft that is defined as "The target certificate MUST match the TLSA record."[11]. This means that in the future it will be easier to deploy self-signed certificates on the Internet.

[8]There currently is discussion on the mailing list whether or not a chain of this length is a valid PKIX chain

[9]`http://www.ietf.org/mail-archive/web/dane/current/msg04099.html`

[10]`http://www.ietf.org/mail-archive/web/dane/current/msg04096.html`

[11]`http://www.ietf.org/mail-archive/web/dane/current/msg04260.html`

Chapter 5

Conclusion

Based on the observations and experiences during the course of this research, the following can be said about DANE.

5.1 Coverage of use cases

The DANE specification covers the three constraints set out in the Use cases section. The usage field covers these cases.

5.2 Implementation

Apart from the issues arising from the lack of clarity of the phrase "pass PKIX validation" combined with usage 2, DANE is a specification that can be implemented and could be the killer application DNSSEC needs for wide-spread deployment.

5.3 Future work

The most pressing matter are the issues described in section 4.5 and are currently being discussed by the working group.

After the specification has become an RFC, SSL/TLS libraries need to implement the secure look up of `TLSA` records and organizations need to deploy `TLSA` records, perhaps along with a revision of their certificate practices. Especially high-profile targets for certificate forgery like banks and software update services can benefit from the added security of DANE.

5.4 Discussion

This project focused purely on the implementation of the DANE specification itself. The following topics were not considered:

- The issues with the PKI system in general
- Possible weaknesses in the specification
- Any of the security issues already mentioned in the specification

Bibliography

[1] D. Cooper et al. *RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. URL: `http://www.ietf.org/rfc/rfc5280.txt`.

[2] R. Barnes. *RFC 6394: Use Cases and Requirements for DNS-Based Authentication of Named Entities (DANE)*. URL: `http://tools.ietf.org/html/rfc6394`.

[3] Peter Eckersley and Jesse Burns. *An Observatory for the SSLiverse*. URL: `www.eff.org/files/DefconSSLiverse.pdf`.

[4] Bruce Morton (Entrust). *Public Key Pinning*. URL: `http://ssl.entrust.net/blog/?p=615`.

[5] Electronic Frontier Foundation. *The Sovereign Keys Project*. URL: `https://www.eff.org/sovereign-keys`.

[6] Danny Groenewegen and Pieter Lange. "Extended Validation using DNSSEC". Universiteit van Amsterdam, 2011.

[7] J. Hodges, C. Jackson, and A. Barth. *HTTP Strict Transport Security (HSTS)*. URL: `http://tools.ietf.org/html/draft-ietf-websec-strict-transport-sec-04`.

[8] P. Hoffman and J. Schlyter. *Using Secure DNS to Associate Certificates with Domain Names For TLS*. URL: `https://datatracker.ietf.org/doc/draft-ietf-dane-protocol/`.

[9] Dan Kaminsky. *Black Ops 2008: It's The End Of The Cache As We Know It*. URL: `http://www.slideshare.net/dakami/dmk-bo2-k8`.

[10] Moxie Marlinspike. *The Convergence Project*. 2011. URL: `http://convergence.io`.

[11] Perspective Project members. *The Perspectives Project*. 2011. URL: `http://perspectives-project.org/`.

[12] Sid Stamm. *HTTP Strict Transport Security has landed!* URL: `http://blog.sidstamm.com/2010/08/http-strict-transport-security-has.html`.

[13] Dan Wendlandt, David Andersen, and Adrian Perrig. "Perspectives: Improving SSH-style Host Authentication with Multi-Path Probing". In: *Proc. USENIX Annual Technical Conference*. Boston, MA, June 2008.

[14] Paul Wouters and Jacob Appelbaum. *sshfp*. URL: `http://www.xelerance.com/software/sshfp/`.

Chapter 6

Addendum

On the fourth of February 2012[1], the working group released the 15th draft of the DANE specification. This chapter discusses the changes introduced compared to the previous draft (which are discussed in this report).

6.1 Usage 3

A new usage (3) has been added, and is defined as:

> “Certificate usage 3 is used to specify a certificate, or the public key of such a certificate, that must match the end entity certificate given by the server in TLS. This usage is sometimes referred to as "domain-issued certificate" because it allows for a domain name administrator to issue certificates for a domain without involving a third-party CA.”

This allows the administrator to issue a self-signed certificate for a service that will be accepted by the TLS client as valid.

6.2 Updated usage 2 description

The description of usage 2 has been updated to better describe how it should be used:

> “Certificate usage 2 is used to specify a certificate, or the public key of such a certificate, that must be used as a trust anchor when validating the end entity certificate given by the server in TLS. This usage allows a domain name administrator to specify a new trust anchor, such as if the domain issues its own certificates under its own CA that is not expected to be in the end users collection of trust anchors.”

6.3 Pass PKIX Validation

The term “pass PKIX validation” is still not well defined in the latest draft. A reference to the PKI Path algorithm in RFC 5280 could help. But the additional information provided with the usage 2 description clears up some confusion on the term. At this moment an amendmend to the draft is discussed to include such a phrase.

6.4 Updates

A few hours after the release of draft 15, `swede` was updated with support for this draft. This allowed the test bed to be updated as well, adding the records and ports mentioned in Table 6.1.

[1]`http://www.ietf.org/mail-archive/web/dane/current/msg04268.html`

Table 6.1: Values for the fields in the `TLSA` record and the certificate used for the corresponding port

Port number	**Usage**	**Selector**	**Matching Type**	**Certificate used**	**Other information**
1522	2	0	1	Private CA	
1523	3	0	1	Self-signed	Usage 3

Appendix A

DNS entries used

```
kiev.practicum.os3.nl. 3600 IN SOA kiev.studlab.os3.nl. hostmaster.os3.nl. 2012012301 14400 3600 604800 3600
dane.kiev.practicum.os3.nl. 3600 IN A 145.100.105.165
dane.kiev.practicum.os3.nl. 3600 IN AAAA 2001:610:158:106a::5
cname1.dane.kiev.practicum.os3.nl. 300 IN CNAME dane.kiev.practicum.os3.nl.
cname2.dane.kiev.practicum.os3.nl. 300 IN CNAME dane.kiev.practicum.os3.nl.
_1500._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 1115 ( 0200003082045430820 2BC020900AB58D24E77AD2AF6300D06092A86
 4886F70D0101050500306C310B3009060355040613024E4C31163014 0603550408130D4E6F6F72642D486F6C6C616E643112301006035504
 071309416D7374657264616D310C300A060355040A13034F53333123 30210603550403131A64616E652E6B6965762E70726163746963756D
 2E6F73332E6E6C301E170D3132303131363136353730335A170D3232 303131333136353730335A306C310B3009060355040613024E4C3116
 30140603550408130D4E6F6F72642D486F6C6C616E64311230100603 5504071309416D7374657264616D310C300A060355040A13034F5333
 31233021060355040313 1A64616E652E6B6965762E70726163746963 756D2E6F73332E6E6C308201A2300D06092A864886F70D0101010500
 0382018F003082018A0282018100E62C84A5AFE59F0A2A6B250DEE68 7AC8C5C604F57D26CEB2119140FFAC38C4B9CBBE8923082E7F81626B
 6AD5DEA0C8771C74E3CAA7F613054AEFA3673E48FFE47B3F7AF987DE 281A68230B24B9DA1A98DCBE51195B60E42FD7517C328D983E26A827
 C877AB914EE4C1BFDEAD48BD25BE5F2C473BA9C1CBBDDDA0C374D0D5 8C389CC3D6D8C20662E19CF768F32441B7F7D14AEA8966CE7C32A172
 2AB38623D008029A9E4702883F8B977A1A1E5292BF8AD72239D40393 37B86A3AC60FA001290452177BF1798609A05A130F033457A5212629
 FBDDB8E70E2A9E6556873C4F7CA46AE4A8B178F05FB319005E1C1C7D 4BD77DFA34035563C126AA2C3328B900E7990AC9787F01DA82F74C3D
 4B6674CCECE1FD4C6EF9E6644F4635EDEDA39D8B0E2F7C8E06DAE775 6213BD3D60831175BE290442B4AFC5AE6F46B769855A067C1097E617
 962529E166F22AEE10DDB981B8CD6FF17D3D70723169038DBFBC1A44 9C8D0D31BC683C5F3CE26148E42EC9BBD4D9F261569B25B53C1D7FC2
 DDFF6B4CAC050203010001300D06092A864886F70D01010505000382 0181002B2ABE063E9C86AC4A1F7835372091079C8276A9C2C5D1EC57
 64DE523FDDABDEAB3FD34E6FE6CBA054580A6785A663595D90132B93 D473929E81FA0887D2FFF78A81C7D014B97778AB6AC9E5E690F6F5A9
 E92BB5FBAB71B857AE69B6E18BDCCB0BA6FCD9D4B084A34F3635148C 495D48FE635903B888EC1DEB2610548EDD48D63F86513A4562469831
 48C0D5DB82D73A4C350A42BB661D763430FC6C8E5F9D13EA1B76AA52 A4C358E5EA04000F794618303AB6CEEA4E9A8E9C74D73C1B0B7BAF16
 DEDE7696B5E2F206F777100F5727E1684D4132F5E692F47AF6756EA8 B421000BE031B5D8F0220E436B51FB154FE9595333C13A2403F9DE08
 E5DDC5A22FD6182E339593E26374450220BC14F3E40FF33F084526B0 9C34250702E8A352B332CCCB0F9DE2CF2B338823B92AFC61C0B6B8AB
 DB5AF718ED8DDA97C298E46B82A01B14814868CFA4F2C36268BFFF4A 591F42658BF75918902D3E426DFE1D5FF0FC6A212071F6DA8BD833FE
 2E560D87775E8EE9333C05B6FB8EB56589D910DB5EA903 )
_1501._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 35 ( 020001EFDDF0D915C7BDC5782C0881E1B2A95AD099FBDD06D7B1F779
 82D9364338D955 )
_1502._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 67 ( 02000281EE7F6C0ECC6B09B7785A9418F54432DE630DD54DC6EE9E3C
 49DE547708D236D4C413C3E97E44F969E635958AA410495844127C04 883503E5B024CF7A8F6A94 )
_1503._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 425 (  020100308201A2300D06092A864886F70D01010105000382018F0030
 82018A0282018100E62C84A5AFE59F0A2A6B250DEE687AC8C5C604F5 7D26CEB2119140FFAC38C4B9CBBE8923082E7F81626B6AD5DEA0C877
 1C74E3CAA7F613054AEFA3673E48FFE47B3F7AF987DE281A68230B24 B9DA1A98DCBE51195B60E42FD7517C328D983E26A827C877AB914EE4
 C1BFDEAD48BD25BE5F2C473BA9C1CBBDDDA0C374D0D58C389CC3D6D8 C20662E19CF768F32441B7F7D14AEA8966CE7C32A1722AB38623D008
 029A9E4702883F8B977A1A1E5292BF8AD72239D4039337B86A3AC60F A001290452177BF1798609A05A130F033457A5212629FBDDB8E70E2A
 9E6556873C4F7CA46AE4A8B178F05FB319005E1C1C7D4BD77DFA3403 5563C126AA2C3328B900E7990AC9787F01DA82F74C3D4B6674CCECE1
 FD4C6EF9E6644F4635EDEDA39D8B0E2F7C8E06DAE7756213BD3D6083 1175BE290442B4AFC5AE6F46B769855A067C1097E617962529E166F2
 2AEE10DDB981B8CD6FF17D3D70723169038DBFBC1A449C8D0D31BC68 3C5F3CE26148E42EC9BBD4D9F261569B25B53C1D7FC2DDFF6B4CAC05
 0203010001 )
_1504._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 35 ( 0201018755CDAA8FE24EF16CC0F2C918063185E433FAAF1415664911
 D9E30A924138C4 )
_1505._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 67 ( 020102D43165B4CDF8F8660AECCCC5344D9D9AE45FFD7E6AAB7AB9EE
 C169B58E11F227ED90C17330CC17B5CCEF0390066008C720CEC6AAE5 33A934B3A2D7E232C94AB4 )
_1506._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 1483 ( 010000308205C4308204ACA003020102021100901 3A9485FF3726155
 064B5ECC30E3B2300D06092A864886F70D0101050500307131 0B3009 0603550406130247423 11B3019060355040813124772656174657220
 4D616E636865737465723110300E0603550407130753616C666F7264 311A3018060355040A1311436F6D6F646F204341204C696D69746564
 31173015060355040313 0E506F736974697665535 34C204341301E17 0D3132303132333030303030305A170D3133303132323233353935 39
 5A305E3121301F060355040B1318446F6D61696E20436F6E74726F6C 2056616C69646174656431143012060355040B130B506F7369746976
 6553534C31233021060355040313 1A64616E652E6B6965762E707261 63746963756D2E6F73332E6E6C308201A2300D06092A864886F70D01
 010105000382018F003082018A0282018100E62C84A5AFE59F0A2A6B 250DEE687AC8C5C604F57D26CEB2119140FFAC38C4B9CBBE8923082E
 7F81626B6AD5DEA0C8771C74E3CAA7F613054AEFA3673E48FFE47B3F 7AF987DE281A68230B24B9DA1A98DCBE51195B60E42FD7517C328D98
 3E26A827C877AB914EE4C1BFDEAD48BD25BE5F2C473BA9C1CBBDDDA0 C374D0D58C389CC3D6D8C20662E19CF768F32441B7F7D14AEA8966CE
 7C32A1722AB38623D008029A9E4702883F8B977A1A1E5292BF8AD722 39D4039337B86A3AC60FA001290452177BF1798609A05A130F033457
 A5212629FBDDB8E70E2A9E6556873C4F7CA46AE4A8B178F05FB31900 5E1C1C7D4BD77DFA34035563C126AA2C3328B900E7990AC9787F01DA
 82F74C3D4B6674CCECE1FD4C6EF9E6644F4635EDEDA39D8B0E2F7C8E 06DAE7756213BD3D60831175BE290442B4AFC5AE6F46B769855A067C
 1097E617962529E166F22AEE10DDB981B8CD6FF17D3D70723169038D BFBC1A449C8D0D31BC683C5F3CE26148E42EC9BBD4D9F261569B25B5
 3C1D7FC2DDFF6B4CAC050203010001A38201E8308201E4301F060355 1D23041830168014B8CA11E9063179DBC394C6E8192ABCBB351631A4
 301D0603551D0E04160414742F06E840267B3DDDEE50BE8AFEC1BCD3 C3CB08300E0603551D0F0101FF0404030205A0300C0603551D130101
 FF04023000301D0603551D250416301406082B0601050507030106 08 2B06010505070302304606 03551D20043F303D303B060B2B06010401
 B23101020207302C302A06082B0601050507020116 1E687474703A2F 2F7777772E706F7369746976657373 6C2E636F6D2F43505330690603
 551D1F04623060302FA02DA02B8629687474703A2F2F63726C2E636F 6D6F646F63612E636F6D2F506F736974697665535 34C43412E63726C
 302DA02BA0298627687474703A2F2F63726C2E636F6D6F646F2E6E65 742F506F736974697665535 34C43412E63726C306B06082B06010505
 070101045F305D303506082B06010505073002862968747470 3A2F2F 6372742E636F6D6F646F63612E636F6D2F506F736974697665535 34C
 43412E637274302406082B06010505073001861868747470 3A2F2F6F 6373702E636F6D6F646F63612E636F6D30450603551D11043E303C82
 1A64616E652E6B6965762E70726163746963756D2E6F73332E6E6C82 1E7777772E64616E652E6B6965762E70726163746963756D2E6F7333
 2E6E6C300D06092A864886F70D01010505000382010100017997E35B CB1316289AB8ED3D520E5267FE992DA236A39C38DCC4CEE41C959378
 2A0A7DAA6402D6BD385E4731C6E0CB1F8360855A8359119FA770F247 EE99DA7698731CCD9D42486F60395B7EBCE7D341164FC137A5927DB8
 14AC0A6CE30A1A53BD317CE004FD277815FCD72E379C5F659A051589 A6DA29B875CC114CFAD2E151A6598D9C76393B14A3C7018C2967C2CC
 26CFC03EEDECC2E9D5510623DABC9721B22F00E5AF96A0DAFEDCF35E E5AE5EC0ECF6C33D04AA632CB6106CA8857B0856E0CAE7F8C2CD99A0
 E4BE8FD3E5A58C621C20B61CEB493B1CA75B15D7B3CD16331F20D759 921A4FB701688681D7D9077F6EBF80F435001485F3A0A97332C8CF )
_1507._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 35 ( 0100016478DAE231CC97A04CF6C9E13FA54D2D44F96FD6C1D28597E8
 8A70167F77D59F )
_1508._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 67 ( 010002629D023AEB206B736197AA4E7930115CE0F217CB4E8F055E5D
 645986D7BB90220A759FF0F3DC28F286C45D076EEAF71C7F9813EEE6 97C85A80F00374884C7655 )
_1509._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 425 ( 010100308201A2300D06092A864886F70D01010105000382018F0030
 82018A0282018100E62C84A5AFE59F0A2A6B250DEE687AC8C5C604F5 7D26CEB2119140FFAC38C4B9CBBE8923082E7F81626B6AD5DEA0C877
 1C74E3CAA7F613054AEFA3673E48FFE47B3F7AF987DE281A68230B24 B9DA1A98DCBE51195B60E42FD7517C328D983E26A827C877AB914EE4
 C1BFDEAD48BD25BE5F2C473BA9C1CBBDDDA0C374D0D58C389CC3D6D8 C20662E19CF768F32441B7F7D14AEA8966CE7C32A1722AB38623D008
 029A9E4702883F8B977A1A1E5292BF8AD72239D4039337B86A3AC60F A001290452177BF1798609A05A130F033457A5212629FBDDB8E70E2A
 9E6556873C4F7CA46AE4A8B178F05FB319005E1C1C7D4BD77DFA3403 5563C126AA2C3328B900E7990AC9787F01DA82F74C3D4B6674CCECE1
 FD4C6EF9E6644F4635EDEDA39D8B0E2F7C8E06DAE7756213BD3D6083 1175BE290442B4AFC5AE6F46B769855A067C1097E617962529E166F2
 2AEE10DDB981B8CD6FF17D3D70723169038DBFBC1A449C8D0D31BC68 3C5F3CE26148E42EC9BBD4D9F261569B25B53C1D7FC2DDFF6B4CAC05
 0203010001 )
_1510._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 35 ( 0101018755CDAA8FE24EF16CC0F2C918063185E433FAAF1415664911
 D9E30A924138C4 )
_1511._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 67 ( 010102D43165B4CDF8F8660AECCCC5344D9D9AE45FFD7E6AAB7AB9EE
 C169B58E11F227ED90C17330CC17B5CCEF0390066008C720CEC6AAE5 33A934B3A2D7E232C94AB4 )
_1512._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 1290 ( 00000030820503308203EBA00302010202104CCD4A9A5B4513218CCF
 902F8B2B5171300D06092A864886F70D0101050500308197310B3009 0603550406130255533110B3009060355040813025554311730150603
 550407130E53616C74204C616B652043697479311E301C060355040A 131554686520555345525452555354204E6574776F726B3121301F06
 0355040B1318687474703A2F2F7777772E7573657274727573742E63 6F6D311F301D0603550403131655544E2D5553455246697273742D48
 61726477617265301E170D3036303931383030303030305A170D3230 30353330313034383385A3071310B30090603550406130247423311B
 3019060355040813124772656174657220D616E6368657374657231 10300E0603550407130753616C666F7264311A3018060355040A1311
 436F6D6F646F204341204C696D69746564311730150603550403130E 506F736974697665535534C20434130820122300D06092A864886F70D
 01010105000382010F003082010A0282010100BD4F79582293C9283E 5211002FC0A9208AD72D551E10E7B87FE2862AA4EC5EE9E492589654
 0DF317BA419E159155EFC2EE002018458326DF20CC3DB3A113310A21 5C798379AB24155C56F0B49598A1DAD21BEA16B5CBB7B0C153F9A446
 DAF02E24B8629E8E8B5E2B5A96A0EA50E988FB282A4D9A9C484F83B6 87AE4FC5C8B5D9FDBE3FD1A79DC62C130DC001C7B370F38F69BBB03C
 10DFEB0900843F6EEFFCE32DB4C75D11CCF7F2F1F6E2E3007E120EDE 8D7D00CA3A3DF672E87925A60816F7AB88FF565E0917C05A82056234
 2B28483297D0840CAC1318DB9E66C7AA148B11694DF109D3BA5DA988 3762D8BF039B9DD7E6057EC26A52AA8D0780EA8E0EF31D0203010001
 A382016E3082016A301F0603551D23041830168014A1725F261B2898 43955D0737D585969D4BD2C345301D0603551D0E04160414B8CA11E9
 063179DBC394C6E8192ABCBB351631A4300E0603551D0F0101FF0404 0302010630120603551D130101FF040830060101FF020101307B0603
 551D1F047430723038A036A0348632687474703A2F2F63726C2E636F 6D6F646F63612E636F6D2F55544E2D5553455246697273742D486172
 64776172652E63726C3036A034A0328630687474703A2F2F63726C2E 636F6D6F646F2E6E65742F55544E2D5553455246697273742D486172
 64776172652E63726C30818606082B0601050507010104 7A3078303B 06082B06010505073002862F687474703A2F2F6372742E636F6D6F64
 6F63612E636F6D2F55544E4164645472757374536572766572436 12E 63727430390608 2B06010505073002862D687474703A2F2F6372742E
 636F6D6F646F2E6E65742F55544E4164645472757374536572766572 43412E637274300D06092A864886F70D01010505000382010100 1DB4
 E7F918485DEDFA5AC31DE3B7EF8615C96C1349160D318C315CE7DA87 648EAF1216A45A2D921A3B3E2165AA17C3957E401CFD14690618CDAC
 31EC6AF1169E89263D5B218DE8E8F9AB3DAA3CCB99F2865F88531505 17E5A8D285A74E497DD6DDC78FCE88BF6505E014B53177EEBD2A86E8
 E8A76A2DDA65198C67E26DF64A856683B6324D9F422317D745416A76 04D4ADB98F6EDCC23EE951F29ED8F37FFB502AF08BFD6F0D22362ECE
 0E46F2DE8FDA3B7D1E93ACFCF4322B0FAB011FA5408FE324991F5DB2 AA0CB9E2A1E7920C904B537D1F28EE563DAF186749DFD51EC2A8982B
 4C4783814C2E44C2EFC863EE8B7B5B31F62661BF791CB0A94EA9C950 7BE2 )
_1513._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 35 ( 000001E8EB600F62046E372DB8180BA38FD9F23F83A8690B910AD184
 58B2255C8D8C52 )
_1514._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 67 ( 000002B1BB6103A8B3A9579A0723978BDB63D1BD956FEB7594BD93E7
 443700906250FCCE4027CD05BC06C8BE5A022F509BB20E51E1A4108C 06DFCA7D29BC6F2DC857B3 )
_1515._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 297 ( 00010030820122300D06092A864886F70D01010105000382010F0030
 82010A0282010100BD4F79582293C9283E5211002FC0A9208AD72D55 1E10E7B87FE2862AA4EC5EE9E4925896540DF317BA419E159155EFC2
 EE002018458326DF20CC3DB3A113310A215C798379AB24155C56F0B4 9598A1DAD21BEA16B5CBB7B0C153F9A446DAF02E24B8629E8E8B5E2B
 5A96A0EA50E988FB282A4D9A9C484F83B687AE4FC5C8B5D9FDBE3FD1 A79DC62C130DC001C7B370F38F69BBB03C10DFEB0900843F6EEFFCE3
 2DB4C75D11CCF7F2F1F6E2E3007E120EDE8D7D00CA3A3DF672E87925 A60816F7AB88FF565E0917C05A820562342B28483297D0840CAC1318
 DB9E66C7AA148B11694DF109D3BA5DA9883762D8BF039B9DD7E6057E C26A52AA8D0780EA8E0EF31D0203010001 )
_1516._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 35 ( 000101DF2D479AC580EEAAACF26940DB1F5D85BD979868F3C89653AC
 460C9061551CCE )
_1517._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 67 ( 0001029462FA867DD1CE27E838737622AB7AF873A2D817A395EC1656
 7664678FD498B4C77984A4A242426A481B5185636712C078C9EA2604 493ACC765C26EAB6133469 )
_1518._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 67 ( 000002B1BB6103A8B3A9579A0723978BDB63D1BD956FEB7594BD93E7
 443700906250FCCE4027CD05BC06C8BE5A022F509BB20E51E1A4108C 06DFCA7D29BC6F2DC857B3 )
_1518._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 67 ( 010002629D023AEB206B736197AA4E7930115CE0F217CB4E8F055E5D
 645986D7BB90220A759FF0F3DC28F286C45D076EEAF71C7F9813EEE6 97C85A80F00374884C7655 )
_1519._tcp.cname1.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 35 ( 020001872E38490EB7BA690926F25C64CD449BB615A0B01BE8259570
 440BE1C4FD298D )
_1520._tcp.cname2.dane.kiev.practicum.os3.nl. 300 IN CNAME _1520._tcp.dane.kiev.practicum.os3.nl.
_1520._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 35 ( 020001445BD841D18D65084E9FE1E38F847B871DAF2CC27A966E2BD5
 9C149356007156 )
_1521._tcp.dane.kiev.practicum.os3.nl. 300 IN TYPE65468 \# 66 ( 090603629D023AEB206B736197AA4E7930115CE0F217CB4E8F055E5D
 645986D7BB90220A759FF0F3DC28F286C45D076EEAF71C7F9813EEE6 97C85A80F00374884C76 )
```

Appendix B

swede outputs

B.1 A sample of full verification output

```
$ ./swede verify -p 1513 dane.kiev.practicum.os3.nl
Received the following record for name _1513._tcp.dane.kiev.practicum.os3.nl.:
Usage: 0 (CA Constraint)
Selector: 0 (Certificate)
Matching Type: 1 (SHA-256)
Certificate for Association: e8eb600f62046e372db8180ba38fd9f23f83a8690b910ad18458b2255c8d8c52
This record is valid (well-formed).
Attempting to verify the record with the TLS service...
Got the following IP: 145.100.105.165
SUCCESS (Usage 0): A certificate in the certificate chain offered by the server matches the one mentioned in the TLSA record and is a CA certificate
The matched certificate has Subject: /C=US/ST=UT/L=Salt Lake City/O=The USERTRUST Network/OU=http://www.usertrust.com/CN=UTN-USERFirst-Hardware
```

B.2 Verification of records using swede

B.2.1 All 18 permutations

```
$ for x in {1500..1517}; do
echo '======> '$x; ./swede verify -p $x -q dane.kiev.practicum.os3.nl
done
======> 1500
SUCCESS (usage 2): A certificate in the certificate chain (including the end-entity certificate) offered by the server matches the TLSA record
======> 1501
SUCCESS (usage 2): A certificate in the certificate chain (including the end-entity certificate) offered by the server matches the TLSA record
======> 1502
SUCCESS (usage 2): A certificate in the certificate chain (including the end-entity certificate) offered by the server matches the TLSA record
======> 1503
SUCCESS (usage 2): A certificate in the certificate chain (including the end-entity certificate) offered by the server matches the TLSA record
======> 1504
SUCCESS (usage 2): A certificate in the certificate chain (including the end-entity certificate) offered by the server matches the TLSA record
======> 1505
```

```
SUCCESS (usage 2): A certificate in the certificate chain (including the end-entity certificate) offered by the server matches the TLSA record
======> 1506
SUCCESS (Usage 1): Certificate offered by the server matches the one mentioned in the TLSA record and chains to a valid CA certificate
======> 1507
SUCCESS (Usage 1): Certificate offered by the server matches the one mentioned in the TLSA record and chains to a valid CA certificate
======> 1508
SUCCESS (Usage 1): Certificate offered by the server matches the one mentioned in the TLSA record and chains to a valid CA certificate
======> 1509
SUCCESS (Usage 1): Certificate offered by the server matches the one mentioned in the TLSA record and chains to a valid CA certificate
======> 1510
SUCCESS (Usage 1): Certificate offered by the server matches the one mentioned in the TLSA record and chains to a valid CA certificate
======> 1511
SUCCESS (Usage 1): Certificate offered by the server matches the one mentioned in the TLSA record and chains to a valid CA certificate
======> 1512
SUCCESS (Usage 0): A certificate in the certificate chain offered by the server matches the one mentioned in the TLSA record and is a CA certificate
======> 1513
SUCCESS (Usage 0): A certificate in the certificate chain offered by the server matches the one mentioned in the TLSA record and is a CA certificate
======> 1514
SUCCESS (Usage 0): A certificate in the certificate chain offered by the server matches the one mentioned in the TLSA record and is a CA certificate
======> 1515
SUCCESS (Usage 0): A certificate in the certificate chain offered by the server matches the one mentioned in the TLSA record and is a CA certificate
======> 1516
SUCCESS (Usage 0): A certificate in the certificate chain offered by the server matches the one mentioned in the TLSA record and is a CA certificate
======> 1517
SUCCESS (Usage 0): A certificate in the certificate chain offered by the server matches the one mentioned in the TLSA record and is a CA certificate
```

B.2.2 Validation of two record for the same name

```
$ ./swede verify -p 1518 dane.kiev.practicum.os3.nl
Received the following record for name _1518._tcp.dane.kiev.practicum.os3.nl.:
Usage: 0 (CA Constraint)
Selector: 0 (Certificate)
Matching Type: 2 (SHA-512)
Certificate for Association: b1bb6103a8b3a9579a0723978bdb63d1bd956feb7594bd93e7443700906250fcce4027cd05bc06c8be5a022f509bb20e51e1a4108c06dfca7d29bc6f2dc857b3
This record is valid (well-formed).
Attempting to verify the record with the TLS service...
Got the following IP: 145.100.105.165
SUCCESS (Usage 0): A certificate in the certificate chain offered by the server matches the one mentioned in the TLSA record and is a CA certificate
The matched certificate has Subject: /C=US/ST=UT/L=Salt Lake City/O=The USERTRUST Network/OU=http://www.usertrust.com/CN=UTN-USERFirst-Hardware
Received the following record for name _1518._tcp.dane.kiev.practicum.os3.nl.:
Usage: 1 (End-Entity Constraint)
Selector: 0 (Certificate)
Matching Type: 2 (SHA-512)
Certificate for Association: 629d023aeb206b736197aa4e7930115ce0f217cb4e8f055e5d645986d7bb90220a759ff0f3dc28f286c45d076eeaf71c7f9813eee697c85a80f00374884c7655
This record is valid (well-formed).
Attempting to verify the record with the TLS service...
Got the following IP: 145.100.105.165
SUCCESS (Usage 1): Certificate offered by the server matches the one mentioned in the TLSA record and chains to a valid CA certificate
The matched certificate has Subject: /OU=Domain Control Validated/OU=PositiveSSL/CN=dane.kiev.practicum.os3.nl
```

B.2.3 Validation of redirected name

```
$ ./swede verify -p 1519 cname1.dane.kiev.practicum.os3.nl
Received the following record for name _1519._tcp.cname1.dane.kiev.practicum.os3.nl.:
Usage: 2 (Trust Anchor)
Selector: 0 (Certificate)
Matching Type: 1 (SHA-256)
Certificate for Association: 872e38490eb7ba690926f25c64cd449bb615a0b01be8259570440be1c4fd298d
This record is valid (well-formed).
```

```
Attempting to verify the record with the TLS service...
Got the following IP: 145.100.105.165
SUCCESS (usage 2): A certificate in the certificate chain (including the end-entity certificate) offered by the server matches the TLSA record
The matched certificate has Subject: /C=NL/ST=Noord-Holland/L=Amsterdam/O=OS3/CN=cname1.dane.kiev.practicum.os3.nl
```

B.2.4 Validation of a redirected name and TLSA record

```
$ ./swede verify -p 1520 cname2.dane.kiev.practicum.os3.nl
Received the following record for name _1520._tcp.cname2.dane.kiev.practicum.os3.nl.:
Usage: 2 (Trust Anchor)
Selector: 0 (Certificate)
Matching Type: 1 (SHA-256)
Certificate for Association: 445bd841d18d65084e9fe1e38f847b871daf2cc27a966e2bd59c149356007156
This record is valid (well-formed).
Attempting to verify the record with the TLS service...
Got the following IP: 145.100.105.165
SUCCESS (usage 2): A certificate in the certificate chain (including the end-entity certificate) offered by the server matches the TLSA record
The matched certificate has Subject: /C=NL/ST=Noord-Holland/L=Amsterdam/O=OS3/CN=cname2.kiev.practicum.os3.nl
```

B.2.5 Validation of an invalid record

```
$ ./swede verify -p 1521 dane.kiev.practicum.os3.nl
Received the following record for name _1521._tcp.dane.kiev.practicum.os3.nl.:
Usage: 9 (INVALID)
Selector: 6 (INVALID)
Matching Type: 3 (INVALID)
Certificate for Association: 629d023aeb206b736197aa4e7930115ce0f217cb4e8f055e5d645986d7bb90220a759ff0f3dc28f286c45d076eeaf71c7f9813eee697c85a80f00374884c76
Error: The TLSA record is invalid.
Usage: invalid (9 is not one of 0, 1 or 2)
Selector: invalid (6 is not one of 0 or 1)
Matching Type: invalid (3 is not one of 0, 1 or 2)
```

B.3 Validation of TLSA records on the DANE mailing list

```
$ ./swede verify www.ulthar.us
Received the following record for name _443._tcp.www.ulthar.us.:
Usage: 1 (End-Entity Constraint)
Selector: 1 (SubjectPublicKeyInfo)
Matching Type: 1 (SHA-256)
Certificate for Association: 62d5414cd1cc657e3d30ea5e6d0136e92306e725413c616a51cab4b852c70a1c
This record is valid (well-formed).
Attempting to verify the record with the TLS service...
Got the following IP: 68.33.77.0
SUCCESS (Usage 1): Certificate offered by the server matches the one mentioned in the TLSA record and chains to a valid CA certificate
The matched certificate has Subject: /description=465261-v52Y4PT9o1XbAqZD/CN=www.ulthar.us/emailAddress=i.grok@comcast.net

$ ./swede verify lp0.eu
Received the following record for name _443._tcp.lp0.eu.:
Usage: 1 (End-Entity Constraint)
Selector: 1 (SubjectPublicKeyInfo)
Matching Type: 2 (SHA-512)
Certificate for Association: 490d884c778e9031d8f1bdfb4b6e7673418bad66cb8115e36ced911ea612b688ae7cc1909bf23391574e41865e41a51e03ecbc18fa6125a5a14c7d2ba5e0cff3
This record is valid (well-formed).
Attempting to verify the record with the TLS service...
Got the following IP: 81.2.80.65
SUCCESS (Usage 1): Certificate offered by the server matches the one mentioned in the TLSA record and chains to a valid CA certificate
```

```
The matched certificate has Subject: /CN=proxima.lp0.eu

$ ./swede verify grepular.com
Received the following record for name _443._tcp.grepular.com.:
Usage: 1 (End-Entity Constraint)
Selector: 0 (Certificate)
Matching Type: 1 (SHA-256)
Certificate for Association: fda20e60d267270d6e009c196b323c33d3eb7bfc8af0e8fe4cf7bf563cb5a55d
This record is valid (well-formed).
Attempting to verify the record with the TLS service...
Got the following IP: 178.79.145.246
SUCCESS (Usage 1): Certificate offered by the server matches the one mentioned in the TLSA record and chains to a valid CA certificate
The matched certificate has Subject: /description=R6zg0XWBOSk1CTp3/C=GB/CN=secure.grepular.com/emailAddress=postmaster@grepular.com
```