Certificate chain resolver

Release 1.0.0

Remco Koopmans

Jan 03, 2021

CONTENTS:

1 Installation **3**

2 CLI Usage **5**

3 API **7**

4 cert_chain_resolver **9**
4.1 cert_chain_resolver package 9

5 Indices and tables **13**

Python Module Index **15**

Index **17**

Resolve / obtain the certificate intermediates of a x509 certificate.

CHAPTER

ONE

INSTALLATION

Pypi package:

```
$ pip install cert-chain-resolver
```

From source:

```
$ git clone git@github.com:rkoopmans/python-certificate-chain-resolver.git

$ pip install python-certificate-chain-resolver
```

CHAPTER

TWO

CLI USAGE

$ cert_chain_resolver certificate.crt > bundle.crt

Or read from stdin

$ cat certificate.crt | cert_chain_resolver > bundle.crt

CHAPTER

THREE

API

Resolve a certificate chain

```
>>> from cert_chain_resolver.api import resolve
>>> with open('cert.pem', 'rb') as f:
>>>     fb = f.read()
>>>     chain = resolve(fb)
>>>
>>> for cert in chain:
>>>   print(cert)
<Cert common_name="cert-chain-resolver.remcokoopmans.com" subject="CN=cert-chain-
↪resolver.remcokoopmans.com" issuer="CN=R3,O=Let's Encrypt,C=US">
<Cert common_name="R3" subject="CN=R3,O=Let's Encrypt,C=US" issuer="CN=DST Root CA X3,
↪O=Digital Signature Trust Co.">
<Cert common_name="DST Root CA X3" subject="CN=DST Root CA X3,O=Digital Signature␣
↪Trust Co." issuer="CN=DST Root CA X3,O=Digital Signature Trust Co.">
```

CHAPTER

FOUR

CERT_CHAIN_RESOLVER

4.1 cert_chain_resolver package

4.1.1 Submodules

4.1.2 cert_chain_resolver.cli module

cert_chain_resolver.cli.**cli**(*file_bytes=None*, *show_details=None*)

cert_chain_resolver.cli.**parse_args**()

4.1.3 cert_chain_resolver.exceptions module

exception cert_chain_resolver.exceptions.**ImproperlyFormattedCert**
Bases: Exception

4.1.4 cert_chain_resolver.models module

class cert_chain_resolver.models.**Cert**(*x509_obj*)
Bases: object

The *Cert* object, which is a convenience wrapper for interacting with the underlying cryptography.x509.Certificate object

Parameters **x509_obj** (cryptography.x509.Certificate) – An instance of cryptography.x509.Certificate

Raises **ValueError** – given type is not an instance of cryptography.x509.Certificate

property ca_issuer_access_location
a URL that contains the CA issuer certificate

Type str

property common_name
Extracted common name from the underlying cryptography.x509.Certificate object

Type str

export(*encoding=<Encoding.PEM: 'PEM'>*)
Export the cryptography.x509.Certificate object"

Parameters **encoding** (cryptography.hazmat.primitives.serialization.Encoding, optional) – The output format. Defaults to Encoding.PEM.

Returns ascii formatted

Return type str

property fingerprint
ascii encoded sha256 fingerprint by calling *get_fingerprint()*

Type str

get_fingerprint(*_hash=<class 'cryptography.hazmat.primitives.hashes.SHA256'>*)
Get fingerprint of the certificate

Parameters **_hash** (cryptography.hazmat.primitives.hashes, optional) – Hasher to use. Defaults to hashes.SHA256.

Returns ascii formatted fingerprint

Return type str

property is_ca
Checks whether the Certificate Authority bit has been set

Type bool

property is_root
Checks whether the certificate is a root

Type bool

property issuer
RFC4515 formatted string of the issuer field from the underlying cryptography.x509.Certificate object

Type str

property not_valid_after
from the underlying cryptography.x509.Certificate object

Type datetime.datetime

property not_valid_before
from the underlying cryptography.x509.Certificate object

Type datetime.datetime

property serial
gets the serial from the underlying cryptography.x509.Certificate object

Type str

property signature_hash_algorithm
gets the signature hashing algorithm name from the underlying cryptography.x509.Certificate object

Type str

property subject
RFC4515 formatted string of the subject field from the underlying cryptography.x509.Certificate object

Type str

property subject_alternative_names
Extracted x509 Extensions from the cryptography.x509.Certificate object

Type list(str)

class cert_chain_resolver.models.**CertificateChain**(*chain=None*)
Bases: object

Creates an iterable that contains a list of *Cert* objects.

Parameters **chain** (*CertificateChain*, optional) – Create a new CertificateChain based on this chain. Defaults to None.

property intermediates
A new *CertificateChain* object with only intermediate certificates

property leaf
in the chain. Also known as the 'leaf'

Type First *Cert*

4.1.5 cert_chain_resolver.resolver module

cert_chain_resolver.resolver.**resolve**(*bytes_cert*, *_chain=None*)
A recursive function that follows the CA issuer chain

Parameters

- **bytes_cert** (*bytes*) – A DER/PKCS7/PEM certificate
- **_chain** (CertificateChain, optional) – Chain to complete. Defaults to None.

Returns All resolved certificates in chain

Return type CertificateChain

4.1.6 cert_chain_resolver.utils module

cert_chain_resolver.utils.**load_ascii_to_x509**(*bytes_input*, *ascii_input*)

cert_chain_resolver.utils.**load_bytes_to_x509**(*bytes_input*)

cert_chain_resolver.utils.**load_der_to_x509**(*bytes_input*)

4.1.7 Module contents

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

C

cert_chain_resolver, 11
cert_chain_resolver.cli, 9
cert_chain_resolver.exceptions, 9
cert_chain_resolver.models, 9
cert_chain_resolver.resolver, 11
cert_chain_resolver.utils, 11

INDEX

C

ca_issuer_access_location() (*cert_chain_resolver.models.Cert property*), 9
Cert (*class in cert_chain_resolver.models*), 9
cert_chain_resolver
    module, 11
cert_chain_resolver.cli
    module, 9
cert_chain_resolver.exceptions
    module, 9
cert_chain_resolver.models
    module, 9
cert_chain_resolver.resolver
    module, 11
cert_chain_resolver.utils
    module, 11
CertificateChain (*class in cert_chain_resolver.models*), 10
cli() (*in module cert_chain_resolver.cli*), 9
common_name() (*cert_chain_resolver.models.Cert property*), 9

E

export() (*cert_chain_resolver.models.Cert method*), 9

F

fingerprint() (*cert_chain_resolver.models.Cert property*), 10

G

get_fingerprint() (*cert_chain_resolver.models.Cert method*), 10

I

ImproperlyFormattedCert, 9
intermediates() (*cert_chain_resolver.models.CertificateChain property*), 11
is_ca() (*cert_chain_resolver.models.Cert property*), 10
is_root() (*cert_chain_resolver.models.Cert property*), 10
issuer() (*cert_chain_resolver.models.Cert property*), 10

L

leaf() (*cert_chain_resolver.models.CertificateChain property*), 11
load_ascii_to_x509() (*in module cert_chain_resolver.utils*), 11
load_bytes_to_x509() (*in module cert_chain_resolver.utils*), 11
load_der_to_x509() (*in module cert_chain_resolver.utils*), 11

M

module
    cert_chain_resolver, 11
    cert_chain_resolver.cli, 9
    cert_chain_resolver.exceptions, 9
    cert_chain_resolver.models, 9
    cert_chain_resolver.resolver, 11
    cert_chain_resolver.utils, 11

N

not_valid_after() (*cert_chain_resolver.models.Cert property*), 10
not_valid_before() (*cert_chain_resolver.models.Cert property*), 10

P

parse_args() (*in module cert_chain_resolver.cli*), 9

R

resolve() (*in module cert_chain_resolver.resolver*), 11

S

self_signed() (*cert_chain_resolver.models.Cert property*), 10
signature_hash_algorithm() (*cert_chain_resolver.models.Cert property*), 10
subject() (*cert_chain_resolver.models.Cert property*), 10

subject_alternative_names()
(*cert_chain_resolver.models.Cert property*), 10