MTAT.07.017
Applied Cryptography

Public Key Infrastructure (PKI)
Public Key Certificates (X.509)

University of Tartu

Spring 2020
Key management

• The hardest problem

• How to obtain the key of the other party?
  • Symmetric key?
    • Confidential and authentic channel needed
  • Asymmetric key?
    • Authentic channel needed

• Trust models:
  • Trust on first use (e.g., SSH)
  • Decentralized model - web of trust (e.g., PGP)
  • Centralized model - trusted third party (e.g., TLS)
Trust on first use (TOFU)

- Used by SSH (encrypted telnet)
- For the first time:

```
$ ssh user@math.ut.ee
The authenticity of host 'math.ut.ee (193.40.37.95)' can't be established.
RSA key fingerprint is SHA256:2x7va2E9JDr1xwRemr5gYQrguFjBGikei9bXD6K44.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'math.ut.ee,193.40.37.95' (RSA) to the list of known hosts.
user@math.ut.ee’s password:
```

```
$ cat ~/.ssh/known_hosts
cs.ut.ee,193.40.36.81 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA2HotFObR9U8MgTE67bGJrLFy4DQ/uVN6ZWNucwt1UJ4+
math.ut.ee,193.40.37.95 ssh-rsa AAAAB3NzaC1yc2EAAAABIwAAAQEA+EuCKTMZU9LYhNqBLfz8KGgqvLv90wiadU0AHv2L1
```

- In the future:

```
$ ssh user@cs.ut.ee
user@math.ut.ee’s password:
```
Trust on first use (TOFU)

• If the key has changed:

$ ssh user@math.ut.ee
@WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! @
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
SHA256:4JlsTx1vbxHYaF6ALHD/dTkX5N6ViZZQtNItAKd04k.
Please contact your system administrator.

• Threat model?
• How to make it more secure?
Web of trust (WOT)
Public key infrastructure (PKI)

Public key certificate binds a public key with an identity

- Main actors:
  - User (subject, subscriber, end-entity)
  - Certificate authority (trusted third party, issuer)
  - Verifier (relying party)

- The passport analogy
PKI use cases

Server authentication

Client authentication

Digital signatures (eIDAS)

Code signing

E-mail (S/MIME)
Certificate Authority (CA)

Certificate Authority – **Trusted** Third Party

Where the trust comes from?

- Software vendors decided on your behalf (Mozilla, Google)
- EU Regulation 910/2014 (eIDAS)

- Root CA – self-signed certificate (trust anchor)
- CA can delegate trust to subordinate/intermediate CAs
- Useful for risk limitation
How to become a CA?

• The goal: profit (Mark Shuttleworth, Thawte (VeriSign), $575 million)
• Get your root CA trusted:
  • Compliance audit (WebTrust, ETSI TS)
  • Ernst & Young or KPMG (15k EUR/year)
  • Liability insurance (required by eIDAS)
  • Insurance industry reluctant (3k EUR/year)
  • Use of Hardware Security Modules (HSM)

http://bugzilla.mozilla.org/show_bug.cgi?id=414520
X.509 certificate

Subject: “John Smith”
Issuer: Verisign, Inc. Root CA

Public key: BC F7 C6 74 F5 32 D0 34 ...
Serial #: 11:21:56:2D:2E
Valid: from 2018.01.01 15:00 to 2020.01.01 15:00
Other data: ...

Signed: CA’s Signature
X.509 certificate

Certificate ::= SEQUENCE {
    tbsCertificate TBSCertificate,
    signatureAlgorithm AlgorithmIdentifier,
    signatureValue BIT STRING }

TBSCertificate ::= SEQUENCE {
    version [0] EXPLICIT Version DEFAULT v1(0),
    serialNumber INTEGER,
    signature AlgorithmIdentifier,
    issuer Name,
    validity Validity,
    subject Name,
    subjectPublicKeyInfo SubjectPublicKeyInfo,
    extensions [3] EXPLICIT Extensions OPTIONAL -- v3(2) only }

Validity ::= SEQUENCE {
    notBefore UTCTime,
    notAfter UTCTime }

Extensions ::= SEQUENCE SIZE (1..MAX) OF Extension
Extension ::= SEQUENCE {
    extnID OBJECT IDENTIFIER,
    critical BOOLEAN DEFAULT FALSE,
    extnValue OCTET STRING }

X.509 certificate

- tbsCertificate – DER structure to be signed by CA
- version – X.509v1 or X.509v3 used
  - X.509 v3 introduces certificate extensions
- serialNumber – unique for every certificate issued by CA
- signature – AlgorithmIdentifier from outer Certificate sequence
- issuer – identity of CA who signed the certificate
- validity – period in which certificate should be assumed valid
- subject – identity of a subject whose public key in certificate
- subjectPublicKeyInfo – subject’s public key
- extensions – optional extensions providing more information
Distinguished Name (DN) in X.509 Certificate

The issuer and subject field is defined as the X.501 type Name:

Name ::= RDNSequence
RDNSequence ::= SEQUENCE OF RelativeDistinguishedName
RelativeDistinguishedName ::= SET OF AttributeTypeAndValue

AttributeTypeAndValue ::= SEQUENCE {
  type OBJECT IDENTIFIER,
  value ANY -- DEFINED BY type  }

- Yet another notation for unique identifiers
- Used in LDAP and related protocols
- Example: CN=John Doe, OU=Helpdesk, O=Burgers Inc., C=US
Distinguished Name (DN) in X.509 Certificate

2  74: SEQUENCE {
4  11:   SET {
6   9:     SEQUENCE {
8   3:       OBJECT IDENTIFIER countryName (2 5 4 6)
13  2:       PrintableString 'US'
       :   }
       :   }
17  18:   SET {
19  16:     SEQUENCE {
21  3:       OBJECT IDENTIFIER organizationName (2 5 4 10)
26   9:       UTF8String 'Burgers Inc.'
       :   }
       :   }
37  20:   SET {
39  18:     SEQUENCE {
41  3:       OBJECT IDENTIFIER organizationalUnitName (2 5 4 11)
46  11:       UTF8String 'Helpdesk'
       :   }
       :   }
59  17:   SET {
61  15:     SEQUENCE {
63  3:       OBJECT IDENTIFIER commonName (2 5 4 3)
68   8:       UTF8String 'John Doe'
       :   }
       :   }

(2 5 4 4) : surname (SN)
(2 5 4 42) : givenName (GN)
(2 5 4 5) : serialNumber
(2 5 4 7) : localityName (L)
(2 5 4 8) : stateOrProvinceName (ST)
(1 2 840 113549 1 9 1) : emailAddress
Certificate extensions (X.509v3 only)

Extensions ::= SEQUENCE SIZE (1..MAX) OF Extension
Extension ::= SEQUENCE {
  extnID OBJECT IDENTIFIER,
  critical BOOLEAN DEFAULT FALSE,
  extnValue OCTET STRING }

• Every extension has its OID
• RFC 5280 defines several standard extensions
• Certificate (path) validation algorithm must handle those

"Each extension in a certificate is designated as either critical or non-critical. A certificate-using system MUST reject the certificate if it encounters a critical extension it does not recognize or a critical extension that contains information that it cannot process. A non-critical extension MAY be ignored if it is not recognized, but MUST be processed if it is recognized."

Certificate extensions (X.509v3 only)

• Key usage – limits the purpose of the key contained in the certificate

KeyUsage ::= BIT STRING {
  digitalSignature     (0),
  nonRepudiation      (1), -- contentCommitment
  keyEncipherment     (2),
  dataEncipherment    (3),
  keyAgreement        (4),
  keyCertSign         (5),
  cRLSign             (6),
  encipherOnly        (7),
  decipherOnly        (8) }

• If extension is not present the key may be used for all purposes

• Extended key usage – indicates a more specific purpose of the key

ExtKeyUsageSyntax ::= SEQUENCE SIZE (1..MAX) OF KeyPurposeId
KeyPurposeId ::= OBJECT IDENTIFIER
id-kp-serverAuth  OBJECT IDENTIFIER ::= { 1 3 6 1 5 5 7 3 1 }
id-kp-clientAuth  OBJECT IDENTIFIER ::= { 1 3 6 1 5 5 7 3 2 }
id-kp-codeSigning OBJECT IDENTIFIER ::= { 1 3 6 1 5 5 7 3 3 }
id-kp-emailProtection OBJECT IDENTIFIER ::= { 1 3 6 1 5 5 7 3 4 }

• Usage must be consistent with the key usage extension
Certificate extensions (X.509v3 only)

- Basic constraints – identifies whether subject is CA
  - For CA certificate identifies maximum subordinate CAs it may have
    
    - If `cA` is TRUE, the key usage extension must be absent or must have `keyCertSign` bit set

- Certificate policies – contains pointer to policy information
  - URL to certificate practice statement (CPS)
  - OID of the CPS document version
  - Explicit notice text
Certificate extensions (X.509v3 only)

- **Subject alternative name**
  - Identifies subject alternatively to the subject name
  - Include email, DNS name, IP addresses, URI, etc.
  - New standards promote use of this extension

- **Authority key identifier and subject key identifier**
  - Uniquely identifies subject and issuer

- **CRL distribution points**
  - Includes URI where CRL is available (HTTP or LDAP)

- **Authority information access**
  - Indicates how to access information about CA services

- **Subject information access**
  - Indicates how to access information about subject

Extensions may include a picture of the subject, attributes, roles, etc.
Use in HTTPS (TLS)

- TLS server certificates – the most popular use case
- What does the browser verify before the connection is considered secure?
  - Certificate signed by a trusted CA
  - Host name in the address bar matches the CN in the certificate
  - Validity date, extensions, etc.
# Server certificate

This certificate has been verified for the following uses:

- SSL Client Certificate
- SSL Server Certificate

**Issued To**

- Common Name (CN): auth.ut.ee
- Organization (O): Tartu Ülikool
- Organizational Unit (OU): <Not Part Of Certificate>

**Issued By**

- Common Name (CN): TERENA SSL CA 3
- Organization (O): TERENA
- Organizational Unit (OU): <Not Part Of Certificate>

**Period of Validity**

- Begins On: September 27, 2016
- Expires On: October 2, 2019

**Fingerprints**

Server certificate
Server certificate

$ openssl x509 -in authutee.crt -text
Version: 3 (0x2)
Signature Algorithm: sha256WithRSAEncryption
Issuer: C=NL, ST=Noord-Holland, L=Amsterdam, O=TERENA, CN=TERENA SSL CA 3
Validity
   Not Before: Sep 27 00:00:00 2016 GMT
   Not After : Oct 2 12:00:00 2019 GMT
Subject: C=EE, ST=Tartumaa, L=Tartu, O=Tartu ulikool, CN=auth.ut.ee
Subject Public Key Info:
   Public Key Algorithm: rsaEncryption
   Public-Key: (2048 bit)
   Modulus:
   Exponent: 65537 (0x10001)
X509v3 extensions:
   X509v3 Basic Constraints: critical
      CA:FALSE
   X509v3 Key Usage: critical
      Digital Signature, Key Encipherment
   X509v3 Extended Key Usage:
      TLS Web Server Authentication, TLS Web Client Authentication
X509v3 Subject Alternative Name:
   DNS:auth.ut.ee, DNS:parool.ut.ee, DNS:passwd.ut.ee
X509v3 CRL Distribution Points:
   Full Name: URI:http://crl3.digicert.com/TERENASSLCA3.crl
Signature Algorithm: sha256WithRSAEncryption
  0c:89:c4:3d:2f:c0:76:d9:92:f2:eb:2f:c0:e8:7f:fd:b0:56:
Certificate authorities

<table>
<thead>
<tr>
<th>Certificate Name</th>
<th>Security Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon</td>
<td></td>
</tr>
<tr>
<td>Amazon Root CA 1</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Amazon Root CA 2</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Amazon Root CA 3</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Amazon Root CA 4</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Amazon</td>
<td>Software Security Device</td>
</tr>
<tr>
<td>AS Sertifiteerimiskeskus</td>
<td></td>
</tr>
<tr>
<td>EE Certification Centre Root CA</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>KLAS3-SK 2016</td>
<td>Software Security Device</td>
</tr>
<tr>
<td>KLAS3-SK 2010</td>
<td>Software Security Device</td>
</tr>
<tr>
<td>KLAS3-SK 2010</td>
<td>Software Security Device</td>
</tr>
<tr>
<td>Atos</td>
<td></td>
</tr>
<tr>
<td>Atos TrustedRoot 2011</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Autoridad de Certificacion Firmaprofesional CIF A62634068</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Autoridad de Certificacion Firmaprofesional CIF A62634068</td>
<td>Builtin Object Token</td>
</tr>
<tr>
<td>Baltimore</td>
<td></td>
</tr>
</tbody>
</table>
Identity verification

- Domain Validation (DV): $20/year $0/year
  Checks whether you control the domain
  
  https://www.pilet.ee/cgi-bin/splususer/splususer.cgi

- Organization Validation (OV): $200/year
  Checks whether you operate the organization
  
  https://www.eesti.ee/et/index.html

- Extended Validation (EV): $500/year
  Checks whether you operate the organization x2
  
  Swedbank AS (EE) https://www.swedbank.ee/private
Domain Validation (DV) vs Organization Validation (OV)
Certificate signing request (CSR)

```bash
$ openssl genrsa -out priv.pem 2048
$ openssl req -new -key priv.pem -out auth.ut.ee.csr
```

You are about to be asked to enter information that will be incorporated into your certificate request.

What you are about to enter is what is called a Distinguished Name or a DN.

There are quite a few fields but you can leave some blank.

For some fields there will be a default value,

If you enter '.', the field will be left blank.

```
-----
Country Name (2 letter code) [AU]: EE
State or Province Name (full name) [Some-State]: .
Locality Name (eg, city) []: .
Organization Name (eg, company) [Internet Widgits Pty Ltd]: .
Organizational Unit Name (eg, section) []: .
Common Name (e.g. server FQDN or YOUR name) []: auth.ut.ee
Email Address []: .
```

Please enter the following 'extra' attributes to be sent with your certificate request

A challenge password []: asdasd
An optional company name []: .

```bash
$ cat auth.ut.ee.csr
-----BEGIN CERTIFICATE REQUEST-----
MIIBZzCB0QIBADAoMQswCQYDVQQGEwJFRTEZMBcGA1UEAwwQd3d3LmFwcGNyeXB0I2Vmz+8IpKax5en8M29CGwuL4elOua6LejVE
```

26 / 38
Certificate signing request (CSR)

```
$ openssl req -in auth.ut.ee.csr -text
Certificate Request:
    Data:
        Version: 0 (0x0)
        Subject: C=EE, CN=auth.ut.ee
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                    Modulus:
                    Exponent: 65537 (0x10001)
        Attributes:
            challengePassword :unable to print attribute
        Signature Algorithm: sha256WithRSAEncryption
```
Certificate signing request (CSR)

```bash
$ openssl req -in auth.ut.ee.csr -outform der -out auth.ut.ee.csr.der
$ dumpasn1 auth.ut.ee.csr.der

0 376: SEQUENCE {
  4 226: SEQUENCE {
    7 1: INTEGER 0
    10 34: SEQUENCE {
      12 11: SET {
        14 9: SEQUENCE {
          16 3: OBJECT IDENTIFIER countryName (2 5 4 6)
          21 2: PrintableString 'EE'
        }
      }
      25 19: SET {
        27 17: SEQUENCE {
          29 3: OBJECT IDENTIFIER commonName (2 5 4 3)
          34 10: UTF8String 'auth.ut.ee'
        }
      }
    }
    46 159: SEQUENCE {
      49 13: SEQUENCE {
        51 9: OBJECT IDENTIFIER rsaEncryption (1 2 840 113549 1 1 1)
        62 0: NULL
      }
      64 141: BIT STRING, encapsulates {
        68 137: SEQUENCE {
          71 129: INTEGER
            00 94 94 DA 30 50 8D ED 4D B9 1F 3C 74 58 C3 BC
        }
        203 3: INTEGER 65537
      }
      208 23: [0] {
        210 21: SEQUENCE {
          212 9: OBJECT IDENTIFIER challengePassword (1 2 840 113549 1 9 7)
          223 8: SET {
            225 6: UTF8String 'asdasd'
          }
        }
      }
      233 13: SEQUENCE {
        235 9: OBJECT IDENTIFIER sha256WithRSAEncryption (1 2 840 113549 1 1 11)
        246 0: NULL
      }
      248 129: BIT STRING
        6C F7 BE 7F 02 42 8F 4B 53 0D 52 4E 9F E8 5C DC
        B0 7D BA B8 01 7D 00 8B FF 88 79 29 6E 55 5D 18
        A1 FE 01 A7 E8 CF 26 64 17 7D B1 E4 A9 00 22 C2
      }
    }
  }
}```
Certificate signing request (CSR)

"A certification request consists of a distinguished name, a public key, and optionally a set of attributes, collectively signed by the entity requesting certification. Certification requests are sent to a certification authority, which transforms the request into an X.509 public-key certificate."

\[
\text{CertificationRequest ::= SEQUENCE } \{
\text{certificationRequestInfo CertificationRequestInfo,}
\text{signatureAlgorithm AlgorithmIdentifier,}
\text{signature BIT STRING}
\}\n\]

\[
\text{CertificationRequestInfo ::= SEQUENCE } \{
\text{version INTEGER v1(0),}
\text{subject Name,}
\text{subjectPKInfo SubjectPublicKeyInfo,}
\text{attributes [0] IMPLICIT Attributes}
\}\n\]

PKCS#10: https://tools.ietf.org/html/rfc2986
Enrollment: Step 1

Note: * - required.

Business Email*

Country *

Do you agree to receive communications from DigiCert about available offers and services?

☑ Yes please, I'd like to hear about offers and services.

First Name*

Last Name*

Company*

Province

Phone Number*

What are you primarily interested in??

Securing transactions & protecting sensitive info online

How many domains are you looking to secure??

1-10

Do you have a shopping cart on your website??

No

What is your timeframe for implementation??

Immediate

Need a quote??

No

Comments

By clicking Continue you agree to DigiCert, Inc. or its affiliates processing your data in accordance with DigiCert’s Privacy Policy.
Enrollment: Step 2

SSL Trial

GeoTrust SSL Trial Enrollment

Enter CSR

After generating your server’s Certificate Signing Request as described in Generate CSR, paste the CSR in the form below. Please make sure that it contains the complete header and footer “BEGIN” and “END” lines exactly as in the example below.

Certificate Signing Request

-----BEGIN CERTIFICATE REQUEST-----
MIIBeDCB4gIBADAIjM0QswCQYDVQQGEwFRTETMBEGA1UEAxwKYXV0aC51dC5lZTCBnzANBgkqhkiG9w0BAQEEFAjQAgwYKCQYEAlJTaMFCN7UZ2HxZwOwMB08HDLNghFYvwv+Caeykd6Cvnyh6A21z7Xu5F3S8Zn9pjZrfe+gl6qW9v0CVYRJgfoypcSHGdstcY6nKHdykotKVgsGM438hPv00xVF9718N0kdPqC7Bln3E4MQeKYeHD3TRFo+pnrStV7yLPbwp5z0CAwEAAQBMBSG5Ib3DQEJBzEIDAZh2Rh2c20dQYJKoZIhvcNAQELBQAAdgYEabPe+fwJc9t7TDVZ0n+hC3LB9urgBfQCL/2h4KW5VXRih/g6nGM8mZBd9r5e5pACLmLbZsZT5OjA9sLWP/3vTUR938sdqmuv3y/KJP+am09a9z+VSlECml+B5qeX0B2YQLaf0Eoup8QHe/arjX+MvdhK1WbVTXMKcibAMG=
-----END CERTIFICATE REQUEST-----
SSL Trial

GeoTrust SSL Trial Enrollment

Verify Server URL

The CSR you generated is designed to work with the following URL:

https://auth.ut.ee

If this is not the correct URL (computed from the Common Name in the CSR), or if any of the CSR Information below is incorrect, then please generate a new CSR and click the Replace CSR button.

Replace CSR

CSR Information

Note: The value for the Common Name must exactly match the name of the server you plan to secure.

Common Name: auth.ut.ee
Organization:
Org. Unit:
Locality:
State:
Country: EE

Continue
Enrollment: Step 4

GeoTrust SSL Trial Enrollment

Approval of your certificate request
The GeoTrust SSL Trial service relies upon the Subscriber or the subscriber authorized administrator to approve all certificate requests for all hosts in the domain. We’ll send an email to the domain contact in the domain’s WHOIS record of each domain listed in your certificate information. To validate control of the domain, the owner of domain or an authorized representative must approve the request.

Select your authorized approvers
We send the approval email to the WHOIS contacts for each domain. In case the domain doesn’t have any WHOIS contacts, we send the approval email to the pre-approved email addresses. You can customize the email recipients for each of your base domains if required.

ut.ee
No WHOIS contacts found for this domain. View subdomains in your order
No of Subdomains: 1

List of authorized approvers
You can select a different email address for this base domain, if required.

Pre-approved email addresses
- admin@ut.ee
- administrator@ut.ee
- hostmaster@ut.ee
- webmaster@ut.ee
- postmaster@ut.ee

Back  Continue
Task: Certificate issuer – 6p

Implement a utility that issues TLS server certificate based on a certificate signing request:

```
$ ./issue_cert.py
usage: issue_cert.py private_key CA_cert csr output_cert

$ ./issue_cert.py UT_priv.pem UT_rootCA.pem appcrypto.ee.csr issued.pem
[+] Issuing certificate for "www.appcrypto.ee"

$ openssl verify -CAfile UT_rootCA.pem -purpose sslserver issued.pem
issued.pem: OK

$ openssl verify -CAfile UT_rootCA.pem -purpose smimesign issued.pem
CN = www.appcrypto.ee
error 26 at 0 depth lookup: unsupported certificate purpose
error issued.pem: verification failed
```

- Must support PEM/DER inputs, PEM output
- Sign subject’s certificate using CA private key
- Use sha256WithRSAEncryption (1.2.840.113549.1.1.11)
- Use CN from subject’s CSR DN (other fields must be ignored!)
Task: Certificate issuer

- Specify any serial you want
- Certificate must be valid for at least ± 3 months (may hardcode)
- Fetch subject’s public key from CSR (subjectPublicKeyInfo)
- Fetch issuer’s distinguished name from CA certificate
- Critical extensions:
  - basic constraints CA:FALSE
  - key usage: digitalSignature
  - extended key usage: id-kp-serverAuth
- Use your own DER encoder and pyasn1 for decoding
Task: Certificate issuer

```bash
$ openssl x509 -in issued.pem -text
Certificate:
  Data:
    Version: 3 (0x2)
    Serial Number: 908108597 (0x3620a335)
  Signature Algorithm: sha256WithRSAEncryption
  Issuer: C = EE, O = University of Tartu, OU = IT dep, CN = UT Root CA
  Validity
    Not Before: Jan 20 12:59:31 2020 GMT
    Not After : Jul 20 12:59:31 2020 GMT
  Subject: CN = www.appcrypto.ee
  Subject Public Key Info:
    Public Key Algorithm: rsaEncryption
      Public-Key: (2048 bit)
        Modulus:
          5c:b7:20:1d:fa:81:95:8d:5a:b0:9e:3e:10:f3:ef:
        Exponent: 65537 (0x10001)
  X509v3 extensions:
    X509v3 Basic Constraints: critical
      CA:FALSE
    X509v3 Key Usage: critical
    Digital Signature
    X509v3 Extended Key Usage: critical
    TLS Web Server Authentication
  Signature Algorithm: sha256WithRSAEncryption
    03:b0:61:34:de:82:8b:9a:3f:00:b0:73:95:27:00:09:11:6a:
```

36 / 38
Task: Hints

• pyasn1 will fail to decode CSR if it contains no attributes (challenge password) since it expects implicit tagging:
  • Make sure your CSR contains challenge password

• pyasn1 can easily encode decoded substructures:
  `encoder.encode(decoder.decode(der)[0][0][5])`

• You may want to implement `asn1_bitstring_der()` which takes byte string as input (padding always 0x00)

• Read ASN.1 definitions or dumpasn1 example certificates to find out DER encoding of certificate and its extensions
  `openssl x509 -inform pem -in cert.pem -outform der -out cert.der`

• For debugging use two windows to compare your dumpasn1 output with reference output
Questions

• What does PKI and X.509 certificates solve?
• Which are the two most important fields in X.509 certificate?
• Who defines trusted CAs for digital signature certificates?
• What is Hardware Security Module useful for?
• What browser checks in a certificate received from the server?
• Who defines trusted CAs for web server certificates?
• How are DV certificates different from OV certificates?
• How does CA verify whether the buyer owns the domain?