



public_key

Copyright © 2008-2020 Ericsson AB, All Rights Reserved
public_key 1.8
June 21, 2020

Copyright © 2008-2020 Ericsson AB, All Rights Reserved

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0> Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. The Initial Developer of the Original Code is Ericsson AB. Ericsson AB, All Rights Reserved.

June 21, 2020

1.2 Public-Key Records

```
| {dNSName, string()}
| {x400Address, string()}
| {directoryName, {rdnSequence, [#AttributeTypeAndValue'{}]]}}
| {ediPartyName, special_string()}
| {ediPartyName, special_string(), special_string()}
| {uniformResourceIdentifier, string()}
| {iPAddress, string()}
| {registeredId, oid()}
| {otherName, term()}
special_string() =
    {teletexString, string()}
    | {printableString, string()}
    | {universalString, string()}
    | {utf8String, binary()}
    | {bmpString, string()}
dist_reason() =
    unused
    | keyCompromise
    | cACompromise
    | affiliationChanged
    | superseded
    | cessationOfOperation
    | certificateHold
    | privilegeWithdrawn
    | aACompromise
OID_macro() =
    ?OID_name()
OID_name() =
    atom()
```

1.2.2 RSA

Erlang representation of **Rivest-Shamir-Adleman cryptosystem (RSA)** keys follows:

id-kp-timeStamping
id-kp-OCSPSigning

Table 2.5: Key Purpose OIDs

1.2 Public-Key Records

```
#'AuthorityKeyIdentifier'{
  keyIdentifier,      % oid()
  authorityCertIssuer, % general_name()
  authorityCertSerialNumber % integer()
}.

#'PrivateKeyUsagePeriod'{
  notBefore, % general_time()
  notAfter   % general_time()
}.

#'PolicyInformation'{
  policyIdentifier, % oid()
  policyQualifiers  % [#PolicyQualifierInfo{}]
}.

#'PolicyQualifierInfo'{
  policyQualifierId, % oid()
  qualifier          % string() | #'UserNotice'{}
}.

#'UserNotice'{
  noticeRef, % #'NoticeReference'{}
  explicitText % string()
}.

#'NoticeReference'{
  organization, % string()
  noticeNumbers % [integer()]
}.

#'PolicyMappings_SEQOF'{
  issuerDomainPolicy, % oid()
  subjectDomainPolicy % oid()
}.

#'Attribute'{
  type, % oid()
  values % [der_encoded()]
}).

#'BasicConstraints'{
  cA, % boolean()
  pathLenConstraint % integer()
}).

#'NameConstraints'{
  permittedSubtrees, % [#'GeneralSubtree'{}]
  excludedSubtrees  % [#'GeneralSubtree'{}]
}).

#'GeneralSubtree'{
  base, % general_name()
  minimum, % integer()
  maximum % integer()
}).

#'PolicyConstraints'{
  requireExplicitPolicy, % integer()
  inhibitPolicyMapping % integer()
}).

#'DistributionPoint'{
  distributionPoint, % {fullName, [general_name()]} | {nameRelativeToCRLIssuer,
  [#AttributeTypeAndValue{}]}
```


2 Reference Manual

The `public_key` application provides functions to handle public-key infrastructure from RFC 3280 (X.509 certificates) and parts of the PKCS standard.

