What Your Mother Didn't Tell You About PEM, DER, PKCS

Eric Norman
University of Wisconsin-Madison

Audience

- I'm nuts
- Some of you might want to bolt
- Who needs to know?
  - Developers
  - Support personnel – diagnose problems
  - System Administrators

Today's Class

- ASN.1, BER, DER Overview
- Usage and formats
  - X.509 certificates
  - PKCS7
  - S/MIME
  - PKCS12
- Tools
  - OpenSSL utilities

ASN.1

- Abstract Syntax Notation (version 1)
- Specifies type and structure
  - Type of value (integer, boolean, character string, etc.)
  - Structure (containment, order, options)
- Does not specify encoding (representation)

ASN.1 Primitive Types

- INTEGER
- BOOLEAN
- OCTET_STRING
- BIT STRING
- OBJECT_IDENTIFIER (OID)
- IA5String (ASCII)
- PrintableString
- UTF8String (UTF8)
- UTCTime (YYMMDDhhmmss)

ASN.1 Structure

- SEQUENCE (ordered collection)
  - record, struct
- SEQUENCE OF (ordered; all same type)
  - vector, array
- SET (unordered collection)
  - no counterpart
- SET OF (unordered; all same type)
  - no counterpart
- CHOICE
  - union

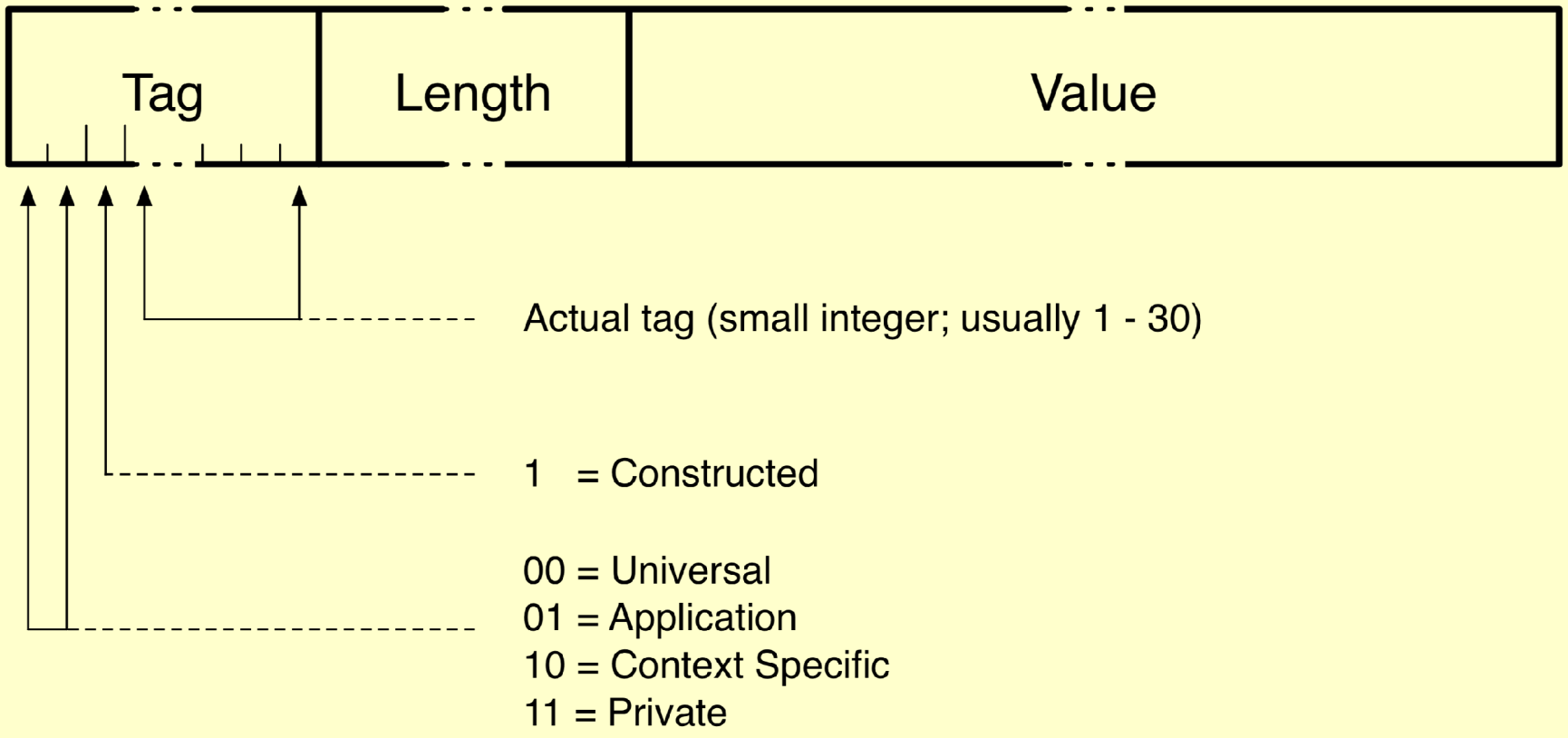
ASN.1 Example

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

TBSCertificate  ::=  SEQUENCE  {
     version         [0]  Version DEFAULT v1,
     serialNumber         CertificateSerialNumber,
     signature            AlgorithmIdentifier,
     issuer               Name,
     validity             Validity,
     subject              Name,
     subjectPublicKeyInfo SubjectPublicKeyInfo,
     issuerUniqueID  [1]  IMPLICIT UniqueIdentifier OPTIONAL,
                          -- If present, version MUST be v2 or v3
     subjectUniqueID [2]  IMPLICIT UniqueIdentifier OPTIONAL,
                          -- If present, version MUST be v2 or v3
     extensions      [3]  Extensions OPTIONAL
                          -- If present, version MUST be v3 --  }
```

Encoding Rules

- BER: Basic Encoding Rules
  - Lots of ways to encode
- DER: Distinguished Encoding Rules
  - Subset of BER
  - Only one way to encode
  - Needed for digital signatures
- Designed for serial transmission
  - “On the wire” specifications

TLV Encoding



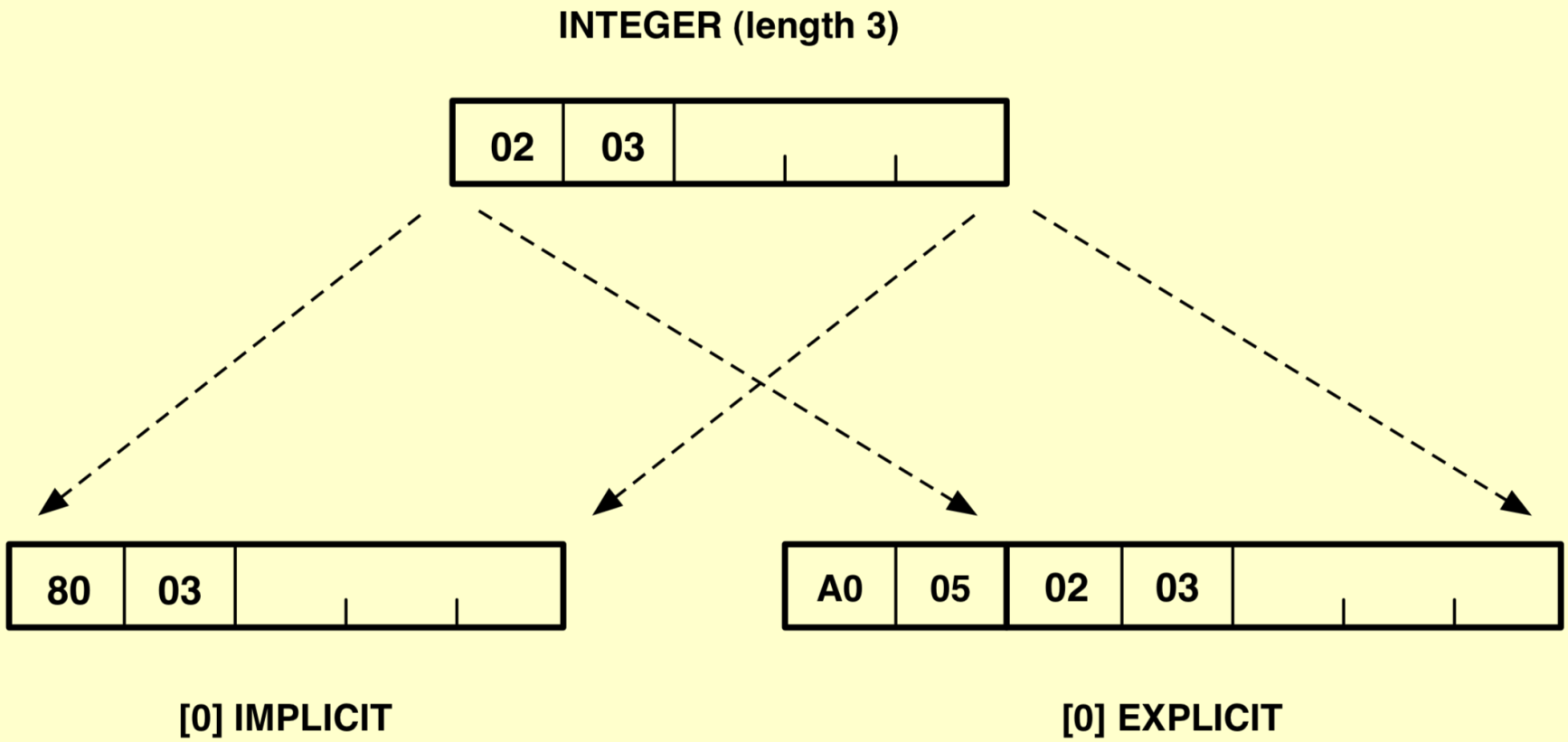
Tags

- Universal
  - Identifies type of value (look at bits)
- Context Specific (and others)
  - Doesn't identify type (if IMPLICIT; read ASN.1)
  - Optional components
  - Choice (discriminated union)

Sample Universal Tags

```
#define V_ASN1_UNIVERSAL                0x00
#define V_ASN1_CONTEXT_SPECIFIC         0x80

#define V_ASN1_CONSTRUCTED              0x20

#define V_ASN1_BOOLEAN                  1
#define V_ASN1_INTEGER                  2
#define V_ASN1_OCTET_STRING             4
#define V_ASN1_OBJECT                   6
#define V_ASN1_UTF8STRING               12
#define V_ASN1_SEQUENCE                 16
#define V_ASN1_IA5STRING                22
#define V_ASN1_UTCTIME                  23
#define V_ASN1_GENERALIZEDTIME          24
```

Implicit/Explicit Tags



You Do Want to Read

A Layman's Guide to a Subset of ASN.1, BER, and DER

by Bert Kaliski
RSA Laboratories

ftp://ftp.rsa.com/pub/pkcs/ps/layman.ps
or consult Google

You Would Rather Not Know

- Sundry integer encodings
  - INTEGERs
  - Tags
  - Lengths
  - Parts of OIDs
- Numbering in BIT STRINGs is backwards
- If you think a camel is appropriate for the perl book, then what animal would you suggest for the committee that invented this stuff?

DER Gotchas

- DEFAULT values must be omitted
- BIT STRINGs can't have trailing 0 bits
- SETs have to be sorted

BER/DER vs. XML

	BER/DER	**XML**
Bandwidth	Abstinent	Profligate
Parsing	Lots of code	Easy
Human readable	Not really	Yes
Editing	Special tool	vi, emacs, etc.
Canonicalization	Easy	Rumored difficult
Future	Legacy	Angle brackets
Encumbered	No	Somewhat

X.509 Certificate

(butchered ASN.1; partial truth)

```
SEQUENCE {
  SEQUENCE {
    serial number: INTEGER  (!!!)
    issuer: Distinguished name
    validity period: SEQUENCE {- -}
    subject: Distinguished name
    public key: SEQUENCE {...}
    extensions: SEQUENCE OF Extension
  }
  signature of issuer: BIT STRING
}
```

ASN.1 Example
(again)

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

TBSCertificate  ::=  SEQUENCE  {
     version         [0]  Version DEFAULT v1,
     serialNumber         CertificateSerialNumber,
     signature            AlgorithmIdentifier,
     issuer               Name,
     validity             Validity,
     subject              Name,
     subjectPublicKeyInfo SubjectPublicKeyInfo,
     issuerUniqueID  [1]  IMPLICIT UniqueIdentifier OPTIONAL,
                          -- If present, version MUST be v2 or v3
     subjectUniqueID [2]  IMPLICIT UniqueIdentifier OPTIONAL,
                          -- If present, version MUST be v2 or v3
     extensions      [3]  Extensions OPTIONAL
                          -- If present, version MUST be v3 --  }
```

Distinguished Name
(truth)

```
SEQUENCE OF {
  SET OF {
    SEQUENCE {
      attribute: OBJECT_IDENTIFIER
      value: UTF8String (or IA5-, ...)
    }
  }
}
```

X.509 V3 Extensions
(truth)

```
SEQUENCE OF {
  SEQUENCE {
    extension ID: OBJECT_IDENTIFIER
    critical: BOOLEAN
    encapsulation: OCTET_STRING
  }
}
```

CrossCertificatePair

```
SEQUENCE {
  [0] Certificate OPTIONAL (I'm the subject)
  [1] Certificate OPTIONAL (I'm the issuer)
}
```

PEM Files

- Syntax

```
-----BEGIN ... keyword-----
Base 64 encoding of BER/DER
-----END ... keyword-----
```

- 7 bit clean
  - Can be transported with electronic mail
  - Can be transported by cutting and pasting
- Can contain comments
- Can have multiple objects in one file

PEM Example

```
-----BEGIN CERTIFICATE-----
MIICwzCCAiygAwIBAgICAzMwDQYJKoZIhvcNAQEEBQAwgbcxCzAJBgNVBAYTAlVT
MRIwEAYDVQQIEwlXaXNjb25zaW4xEDAOBgNVBAcTB01hZGlzb24xKDAmBgNVBAoT
H1VuaXZlcnNpdHkgb2YgV2lzY29uc2luLU1hZGlzb24xKzApBgNVBAsTIkRpdmlz
aW9uIG9mIEluZm9ybWF0aW9uIFRlY2hub2xvZ3kxKzApBgNVBAMTIlVXMS0yMDAz
MTIxOCBNaWRkbGV3YXJlIFRlc3RpbmcgQ0EwHhcNMDMxMjE5MDQwOTQ4WhcNMDcw
NDAzMDQwOTQ4WjCBxzELMAkGA1UEBhMCVVMxEjAQBgNVBAgTCVdpc2NvbnNpbjEQ
MA4GA1UEBxMHTWFkaXNvbjEoMCYGA1UEChMfVW5pdmVyc2l0eSBvZiBXaXNjb25z
aW4tTWFkaXNvbjErMCkGA1UECxMiRGl2aXNpb24gb2YgSW5mb3JtYXRpb24gVGVj
aG5vbG9neTEUMBIGA1UEAxMLRXJpYyBOb3JtYW4xJTAjBgkqhkiG9w0BCQEWFmVq
bm9ybWFuQGRvaXQud2lzYy5lZHUwXDANBgkqhkiG9w0BAQEFAANLADBIAkEAp/27
x8st0GPRUulw5AIOOausOvWw9/B3W4XRVrxsk3Yfn2jW4K+YarQKv4rjXJb4BRIl
PGBzx4KhVa0umcv1rQIDAQABoxAwDjAMBgNVHRMBAf8EAjAAMA0GCSqGSIb3DQEB
BAUAA4GBAA/3rfNggPFRblZMv7UufOrxGxK9QkZAULwbpboYVCophRUM//aRuGO4
s/v/cySA91GLjLl8giPWW7Z3JkIE1Cv+G3ycM83Ygtlu1yhB26/JBvC0jg4hl+c1
g1v32RR+E+pESbfSIYoKoJKBX2WElm0IBlLpDrRL+mUhJmcWMzx1
-----END CERTIFICATE-----
```

PEM? CER? CRT?

- Just conventions; not really standards
- .PEM: Base64 format (always?)
  - See keywords for kind of object
  - Can be multiple objects
- .DER: Binary BER/DER encoding (always?)
  - Only one per file
- .CER, .CRT: Can be either (sometimes?)
  - Some software figures it out (i.e. makes a good guess)

PKCS

- Public Key Cryptography Standards
- Produced by RSA Labs
- Specifies format of objects used during public key operations
- Language is ASN.1
  - This is not XML, folks
- Implemented in RSAREF and BSAFE libraries
- Standards from IETF PKIX working group are a superset and generally compatible

PKCS1 RSA Keypair

```
Private-Key: (512 bit)
modulus:
    00:cc:03:09:93:46:3b:67:5e:07:c0:7e:d4:19:b9
    4d:24:8b:d1:53:77:47:e5:cb:b5:ec:21:ca:26:7e
    22:26:bb:42:6b:3a:e7:3b:af:6b:74:29:dc:22:c0
    8f:c1:b6:39:d4:72:d4:59:1e:96:2c:65:e4:64:9a
    7f:80:8b:e8:1d
publicExponent: 65537 (0x10001)
privateExponent:
    30:e9:09:82:a5:73:d8:74:52:a7:73:c0:a5:ea:26
    f4:7c:10:d3:51:e5:8a:d2:2b:eb:50:ae:86:4b:f7
    24:11:93:2e:ee:f6:6e:45:77:bf:ca:4d:5b:6f:f6
    ff:f2:bd:3d:7b:b5:39:c5:c9:5f:9c:18:28:8a:29
    32:7d:9e:81
prime1:
    00:f3:61:a0:15:1b:36:13:3e:1a:89:c2:cf:c8:66
    b9:45:3c:47:34:55:c6:7d:01:c8:f1:cf:4f:14:a4
    e0:d9:a7
prime2:
    00:d6:96:dd:78:7e:63:de:a4:8f:5e:45:ea:38:c2
    10:4a:e1:47:79:9a:1b:0b:85:c4:2f:82:7c:a8:f5
    1a:e0:9b
exponent1:
    00:c6:5d:50:63:43:7d:6c:6b:96:a7:a7:7f:40:d1
    f0:ab:2c:79:00:7f:d4:ba:38:45:36:48:f8:34:64
    39:db:4b
exponent2:
    00:c2:05:f6:89:93:fd:c8:b9:11:c8:33:7e:eb:82
    cc:28:68:38:b0:02:5e:a1:b4:79:06:5b:fd:4a:e7
    13:3e:31
coefficient:
    00:a5:72:28:27:b7:54:a5:9d:1d:20:15:5b:42:ac
    a6:76:a0:a6:0b:48:4d:b2:4c:0a:f7:83:78:c8:b5
    36:55:aa
```

Needs password based encryption

PKCS1 Private Key (PEM Format?)

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,1304AC2997AE5F62

WQjqC3A+NaqjemXh1YPcvrrrKT9ED45Mctq3Yw+N23p3M5vKz2aIk9Ji7EmCTR8v
hwToXjpbODMhARR6WS841Gn6sMCNbtbqY1Aov2c+ZVvblhhelHFa+SbQROArs0YR
rFW/EKAl+No4QBKeH6uF7JtOf1ozA9A+ck8HfnWxsYIiF04zLNnsZT3F4350ySac
a2M41ZTdTU380pA7xU8iPE4vsaJWh27rBpoC9ELjDSzYgPTtsQYddLn+//TxEHTU
ID4sVWhnp7wsvuOX1iQr++rRWzKgc463QiXngGocKswRjkSIaXN1n2EN7yaHQpGV
qL5+k1nqoU70ZbCVMr7JH2amMQAaP6IKskpE0W2WxfSE1u180lqQIT8MlWmvwalk
kZRx9pLu+IN/NwcAZjXa17DyImlFWcPHKOf6c+V+EC5xzWSUA7Uutw==
-----END RSA PRIVATE KEY-----
```

Someone cheated !

PKCS8 Private Key

```
-----BEGIN ENCRYPTED PRIVATE KEY-----
MIIBgTAbBgkqhkiG9w0BBQMwDgQIqNYWMQBs/VYCAggABIIBYN01SzxggpPDD/TN
Z/SF10LL3WRLuqyexqNYccNCr6L8dZ675I/GVxm8OafP9Uyltpflyx L3nOuwaE0N
G4v3DcORHtGkZVmI0vdwMStosf94AlygeGgMVBgkwyhF+Z14hM11NMrjamqCE4bf
+GP0ThzPKyTMPR/X05ECl0PXMGRrW7W/6uyivYgjAHiwp8NmlJoRibVyeSAnaQSf
ii4ghPYzWD7i2OZfPevqYZrUJPU1LUbyydLCKRBHa6QgH+Zn081wNtlfQOT6sRyD
NTnocVJ86ImCe1+aFng2JQj2WUcKySn0QFdnCcYyIvFLSvpBsXjOK+R/L5qv4dnZ
O1bXQb2K0fOCYwuwakJwzt0zqAuaFYn8RUrfkmULWEC6GRcxLrwGL88efNd1On3W
lWWqypv6CqAH0Wuo6yWtUC1FmfjRvF3Z92PglH3mN6zw8Q6pPGcERkHlt8AAjAQB
F2adQOE=
-----END ENCRYPTED PRIVATE KEY-----
```

S/MIME

- IETF Standard for “secure electronic mail”
- Digital signatures
  - Need canonical form of message to be signed
- Encryption
- Other information for recipient
  - Certificates for verification
  - Sender's public encryption key (certificate)
  - Sender's cryptographic algorithms

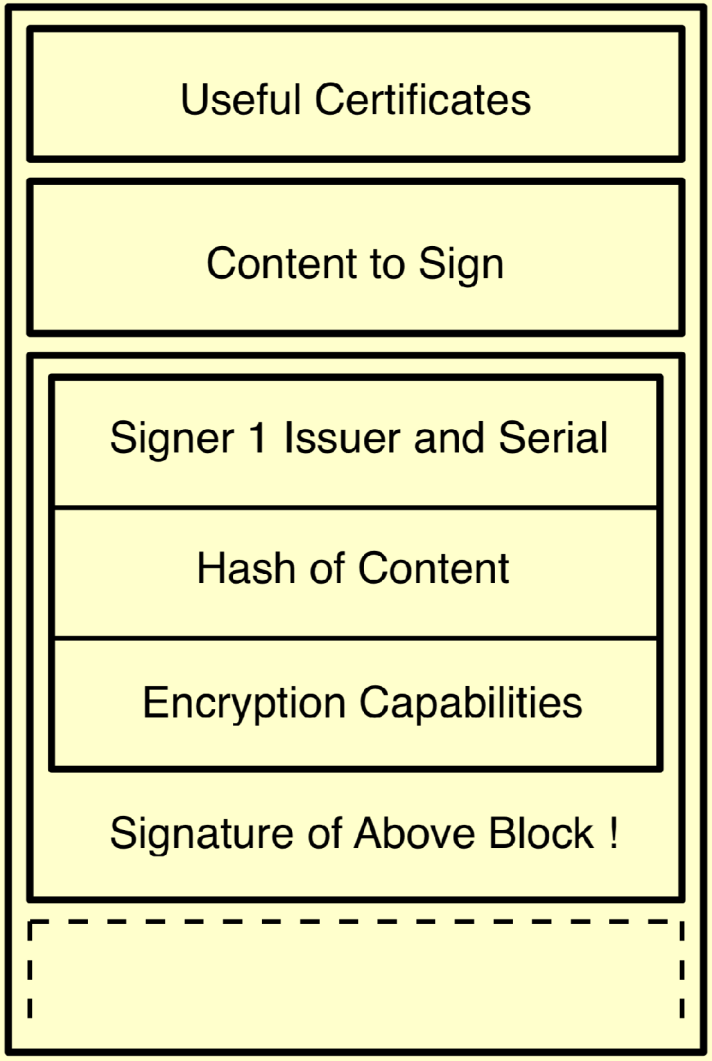
S/MIME
(Signed)

```
From: Eric Norman <ejnorman@doit.wisc.edu>
MIME-version: 1.0
Content-type: multipart/signed; protocol="application/pkcs7-signature";
 boundary=Apple-Mail-3-2162327; micalg=sha1

--Apple-Mail-3-2162327
Content-Transfer-Encoding: 7bit
Content-Type: text/plain; charset=US-ASCII; format=flowed

Message text

--Apple-Mail-3-2162327
Content-Transfer-Encoding: base64
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Disposition: attachment; filename=smime.p7s

MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIIGQzCCAsMw
ggIsoAMCAQICAgMzMA0GCSqGSIb3DQEBBAUAMIG3MQswCQYDVQQGEwJVUzESMBAGA1UECBMJV2lz
 ... snip ...
icLcyxUobN5sT+ttMbm1S6Q+6wAAAAAAA==

--Apple-Mail-3-2162327--
```

PKCS7

- Add cryptographic enhancements to data
  - Signatures
  - Encryption
- Standard does have a type to both sign and encrypt, but it's not used
  - Nesting is used to do both
  - Usually: sign first, then encrypt result
  - Suffers from Davis attack no matter what
- IETF's cryptographic message syntax (CMS) is superset
- Designed for one pass processing

PKCS7
(Signed)



PKCS7
(Signed)

- Three parts; all are optional
  - Certificates
  - Content
  - Signature (with signer information)
- Include all three: opaque signing
- Omit content: detached signature
- Only certificates: “certs only”
  - Used for set/list/chain of certificates
  - File extension = .p7c (or .p7b)

S/MIME

- PKCS7 is not just for electronic mail
- S/MIME is standard for inclusion of PKCS7 objects as MIME “attachments”
- Content-type
  - Multipart/Signed
  - Application/PKCS7-Signature
  - Application/PKCS7-MIME
- Content-transfer-encoding: base64 is not the same thing as PEM format (no -----)

Detached Signature

- Most common for S/MIME mail messages
- Content is readable
- Content-type = Multipart/Signed
  - Part 1 = Text/Plain, Text/HTML, etc.
  - Part 2 = Application/PKCS7-Signature
- File extension (part 2) = .p7s
- Detached content is often mangled by MTAs (like mailing list software)

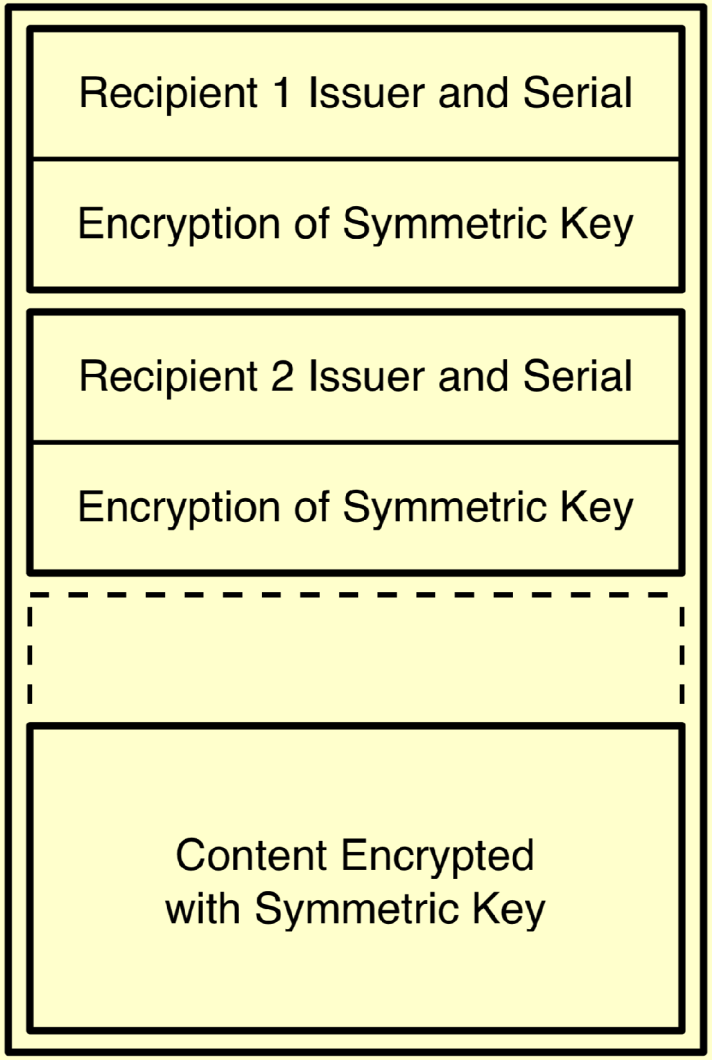
Opaque Signature

- Content is not readable (all base64)
- Content type = Application/PKCS7-MIME
  - (Just one part)
- File extension = .p7m
- Generally not mangled in transit

S/MIME
(Signed)

```
From: Eric Norman <ejnorman@doit.wisc.edu>
MIME-version: 1.0
Content-type: multipart/signed; protocol="application/pkcs7-signature";
 boundary=Apple-Mail-3-2162327; micalg=sha1

--Apple-Mail-3-2162327
Content-Transfer-Encoding: 7bit
Content-Type: text/plain; charset=US-ASCII; format=flowed

Message text

--Apple-Mail-3-2162327
Content-Transfer-Encoding: base64
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Disposition: attachment; filename=smime.p7s

MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIIGQzCCAsMw
ggIsoAMCAQICAgMzMA0GCSqGSIb3DQEBBAUAMIG3MQswCQYDVQQGEwJVUzESMBAGA1UECBMJV2lz
 ... snip ...
icLcyxUobN5sT+ttMbm1S6Q+6wAAAAAAA==

--Apple-Mail-3-2162327--
```

PKCS7
(Encrypted)



S/MIME
(Encrypted or Opaque Signature)

```
To: Eric Norman <ejnorman@doit.wisc.edu>
MIME-version: 1.0
Content-type: application/pkcs7-mime; name=smime.p7m
Content-transfer-encoding: base64
Content-disposition: attachment; filename=smime.p7m

MIAGCSqGSIb3DQEHA6CAMIACAQAxggEZMIIBFQIBADCBvjCBtzELMAkGA1UEBhMCVVMxEjAQBgNV
BAgTCVdpc2NvbnNpbjEQMA4GA1UEBxMHTWFkaXNvbjEoMCYGA1UEChMfVW5pdmVyc2l0eSBvZiBX
aXNjb25zaW4tTWFkaXNvbjErMCkGA1UECxMiRGl2aXNpb24gb2YgSW5mb3JtYXRpb24gVGVjaG5v
 ... snip ...
xAP/MXft3FP8AbeQLSW8rhnAvBz8b3JNE34QLLTV7UXr+9C1vccbDDEYke10HjvTtqVO67FmcL7U
zE5bSBfJjAyjWPypIHAlYNMuFdxedVTHKkaWRrzPb0QqaV9KXrWKRmn2yOGH5BLrRq1TwuohcAQI
pJao+fFIHBIAAAAAAAAAAAA
```

PKCS12

- Purpose is transport or backup
  - Transport = export, copy file, import
- Bundle contains:
  - Private key encrypted with password
  - Associated certificate
  - Other certificates (chain through trust achor)
- Private key must be extractable for export
- File extension = .p12 or .pfx
- Always binary (no PEM)

File Extensions

- Base64
  - .pem: base64 (see keyword)
- Base64 (usually, but might be binary)
  - .cer, .crt: certificate
  - .csr: certificate signing request (PKCS10)
  - .crl: certificate revocation list
- Binary
  - .p7s: PKCS7 signature (detached)
  - .p7m: PKCS7 (encrypted or opaque signature)
  - .p7c, .p7b: PKCS7 certificates only
  - .p12, .pfx: PKCS12 export/import bundle

OpenSSL “Certificate Store”

- Memory
  - Load with `...file` <filename.pem>
  - Multiple certificates; PEM format
- Directory (UNIX, not LDAP)
  - Load with `...path` <directory name>
  - One certificate per file; PEM format
  - Indexed by hash of subject DN; e.g. 1AD4BB3E.0 is symbolic link to PEM file
  - Rarely useful (maybe with stunnel)
  - Create with `tools/c_rehash`

OpenSSL Utilities

- `openssl` <utility> <arguments>...
- Argument order is never important
- Will prompt for passwords when appropriate
  - `-passin` is for scripts
- `openssl -help` to list utilities
- `openssl` <utility> `-help` for specific usage
- There's also man pages
- Just `openssl` gives you a shell
  - Don't have to type “openssl” in front of everything

OpenSSL Utilities
(Common Arguments)

- `-in` <filename>
  - Standard input if omitted
  - `-inform DER` or `PEM` (default)
- `-out` <filename>
  - Standard output if omitted
  - `-outform DER` or `PEM` (default)
  - No overwrite warning
- `-noout -text`
  - Human readable listing

Useful OpenSSL Utilities

- `enc` to decode base64
- `asn1parse` to examine BER/DER
- `x509` to examine certificates
- `dgst` to get fingerprints
- `smime` to disassemble mail messages
- `pkcs7` to extract certificates from .p7s, .p7c
- `pkcs12` to assemble/disassemble .p12
- `s_client` to fetch server certificates

Just Looking

- `dumpasn1` (Peter Gutmann)
  - BER/DER (binary) format only
  - Translates more OIDs
  - Shows encapsulations
- `asn1parse` (OpenSSL utility)
  - PEM format (default; `-inform DER` for binary)
  - Translates some OIDs
  - No encapsulations

4 Fundamental Operations

- Sign
  - Sender uses own private key
- Verify
  - Recipient uses public key of sender
  - Does SSL server know its private key?
- Encrypt
  - Sender uses public key of recipient
- Decrypt
  - Recipient uses own private key

Verification: Murphy Speaks

- No trust anchor
  - Check 3 boxes (Netscape/Mozilla)
- Missing intermediate CA certificate
  - Sender should provide
  - Recipient can provide
- Mangled detached content
  - Mailing lists are notorious
- Unknown extension (rare)
- User can't do hash algorithm

Murphy on Decryption

- Don't have private key
  - Key is “identified” by issuer and serial
  - Same key, different certificates doesn't help
  - Different machines (use PKCS12)
  - Key change (don't delete old key or certificate)
- Vendor didn't read standards
  - How does sender know which public key to use?
  - Recipient can't do symmetric cipher
- Can sender recover cleartext?

The End