Enveloped JSON Signatures

Although XML Signatures are extremely flexible they come at a price: limited interoperability and mobile platform support. The case for XML has also been considerably weakened by REST which more or less has replaced SOAP for web-based systems used by for example Google and Facebook. In REST-based systems the returned data is usually in JSON format.

Converting to JSON

Due to the above I felt a need to convert the currently XML-based KeyGen2 system to JSON. However, there is currently no counterpart to XML DSig's enveloped signatures which made me develop such a system. It turned out that less than 3,000 lines of Java code were required to *Encode*/*Decode* and *Sign/Verify* JSON data:

https://code.google.com/p/openkeystore/source/browse/#svn%2Flibrary%2Ftrunk%2Fsrc%2Forg%2Fwebpki%2Fjson

I can now safely retire my 200,000+ lines Android port of Xerces ☺

Things that make JSON signatures simpler than XML DSig include:

- No attributes that must be sorted
- No namespaces
- No defaults
- No SOAP envelopes
- No WS-Security framework

Obviously there are complex systems that live or die by the use of XML DSig and XML Schema but KeyGen2 is not one of them…

The JSON parser mentioned also does a pretty good job for supporting conformance checking with intended messages though registered message types as well through strict property order and reference checks.

Sample Signature

```
{
  "TestSignatures":
    {
      "@jmns": "http://example.com/signature",
      "Now": "2013-08-30T07:56:08+02:00",
      "HRT":
        {
          "RT1": "67",
          "YT":
            {
              "HTL": "656756#",
              "INTEGER": -689,
              "Fantastic": false
            },
          "er": "33"
        },
      "ARR": [],
      "BARR":
        [{
           "HTL": "656756#",
           "INTEGER": -689,
           "Fantastic": true
         },
         {
             "HTL": "656756#",
             "INTEGER": -689,
             "Fantastic": false
           }],
        "ID": "lADU_sO067Wlgoo52-9L",
        "STRINGS": ["One","Two","Three"],
        "EscapeMe": "A\\\n\"",
        "Intra": 78,
        "EnvelopedSignature":
          {
            "SignatureInfo":
              {
                "Algorithm": "http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256",
                "Reference":
                  {
                    "Name": "ID",
                    "Value": "lADU_sO067Wlgoo52-9L"
                  },
                "KeyInfo":
                  {
                    "SignatureCertificate":
                      {
                        "Issuer": "CN=Demo Sub CA,DC=webpki,DC=org",
                        "SerialNumber": 1377713637130,
                        "Subject": "CN=example.com,O=Example Organization,C=US"
                      },
                    "X509CertificatePath":
                      [

"MIIClzCCAX+gAwIBAgIGAUDGIccKMA0GCSqGSIb3DQEBCwUAMEMxEzARBgoJkiaJk/IsZAEZFgNvcmcxFjAUBgoJkiaJk/IsZA
EZFgZ3ZWJwa2kxFDASBgNVBAMTC0RlbW8gU3ViIENBMB4XDTEyMDEwMTAwMDAwMFoXDTIwMDcxMDA5NTk1OVowQjELMAkGA1UEB
hMCVVMxHTAbBgNVBAoTFEV4YW1wbGUgT3JnYW5pemF0aW9uMRQwEgYDVQQDEwtleGFtcGxlLmNvbTBZMBMGByqGSM49AgEGCCqG
SM49AwEHA0IABECkenu7FrOpy7J2FGeO1vrtseQqJT2GsaExxK5UVKe1zhFXjF+V8OFjv/FdM9fqdqwkP/YUnx5epvvHh/+/cQW
jXTBbMAkGA1UdEwQCMAAwDgYDVR0PAQH/BAQDAgP4MB0GA1UdDgQWBBR4YF2UOnLWDhOPLLuXYZum7xLMajAfBgNVHSMEGDAWgB
RZXCF2vVvvaakHecbUVh7jS1yIVTANBgkqhkiG9w0BAQsFAAOCAQEAjBuZK2TcDhib12DSW8rW3kyjfQ3iYtjNSVd7vJ5jyI+0O
YQ/NlhN4vVJx7Z02vnrBxv1Nh9swgT5Jpw0724KawGC4P+/bUEvKVz89tPMl9DaV98yQ2YN4cBfhcW3FpAoI4dzBbCzfEplsh9E
k7VxuIgwPozl0AdqOmTjZ3hh54ApSq/PMwENDyCEzD6bvrCrqCjgWSYIQUIvQ7LfO2HAlEE9DcoV4mSl/8uiQ05hRdGmNYUHZVU
ua0HHX1h/nAS+IcS6/EDd89kEGrL3M92a5wqnIQvDLO2NBCXhHSxoPVyBzv0lIgaO0ixD+q5P2OszRBYG3uk9W/uNIHdoyQn19w
==",

"MIIDZjCCAk6gAwIBAgICAMgwDQYJKoZIhvcNAQELBQAwRDETMBEGCgmSJomT8ixkARkWA29yZzEWMBQGCgmSJomT8ixkARkWBn
dlYnBraTEVMBMGA1UEAxMMRGVtbyBSb290IENBMB4XDTA1MDcxMDEwMDAwMFoXDTI1MDcxMDA5NTk1OVowQzETMBEGCgmSJomT8
ixkARkWA29yZzEWMBQGCgmSJomT8ixkARkWBndlYnBraTEUMBIGA1UEAxMLRGVtbyBTdWIgQ0EwggEiMA0GCSqGSIb3DQEBAQUA
A4IBDwAwggEKAoIBAQCQI1w6lq0AsHbWMes8i/UGBeVQnlbzL2N8VyLjkbT3HXNHPUTjWWQElhoRzFA2xPaH++V/ecgr2Dkievr
m+B5yIsdAL4oWWmgZ9KVMSfOrl5jy843p6AA55CHlP4j8v1uU1SpexIMUegDcNPlBwSRc0PPX+uqQ1STRg0kUgi4Bap7U5IRxTv
p06adFXU4Bjr85ML7VZ3j+164t6mLnwF5RChJMlO7aVuz6TwxnWqeZytjFOei742dgbX9SHPVvytLtbFp4V/VFoEhaOXLZiOudP
vpVwVdlfgE0AtiGHEWrfA74BU5XhME6UXzjcl3y3Ic304YGymo2jvmOwBki5wb3AgMBAAGjYzBhMA8GA1UdEwEB/wQFMAMBAf8w
DgYDVR0PAQH/BAQDAgEGMB0GA1UdDgQWBBRZXCF2vVvvaakHecbUVh7jS1yIVTAfBgNVHSMEGDAWgBRaQnES9aDCSV/XOklJczx
nqxI4/DANBgkqhkiG9w0BAQsFAAOCAQEAMlPdBaZ/+AMDfFYI9SLQenx0/vludp0oN9BSDe+mTfYNp5nS131cZRCKMAR3g/zzgk
ULu022xTJVsXfM1dsMYwEpGZp+GAvrlmRO6IathHW4aeo0QpaygOgfquQNYgS3Z8OJRSUDGnoY65g50dgvl1+ASbZX/r0/fNANL
zXt/cnf0VXPrWdqvhuUSO561TsbTYg4qzcyDRV5vpjoUAxjFna06TJkeZR/OYMMcTtPRJON3/bMvzp7MFoL20PRPxu8nnqxwLWN
zoQCkExS2yWHq1YDNNL4C/PIuyC/2IUbbPuwNp8ir3MVDBq4QwuXbw6xFvbPsxOmZyH10xvpsnmokg==",

"MIIDZjCCAk6gAwIBAgIBATANBgkqhkiG9w0BAQsFADBEMRMwEQYKCZImiZPyLGQBGRYDb3JnMRYwFAYKCZImiZPyLGQBGRYGd2
VicGtpMRUwEwYDVQQDEwxEZW1vIFJvb3QgQ0EwHhcNMDIwNzEwMTAwMDAwWhcNMzAwNzEwMDk1OTU5WjBEMRMwEQYKCZImiZPyL
GQBGRYDb3JnMRYwFAYKCZImiZPyLGQBGRYGd2VicGtpMRUwEwYDVQQDEwxEZW1vIFJvb3QgQ0EwggEiMA0GCSqGSIb3DQEBAQUA
A4IBDwAwggEKAoIBAQCMR4ROlJJyBUzQ92XDSzFuxiMjwFqsrKXuIksIXMypjg2QZF4PzyQ0pu32/LVKuoaqbT+bkRKFdpUvMKh
zGQ3rMAthTUkhXpFJN5Bk2LTcGXoE0B9wPKn4C3cxbUMEtT94m8PKIjRoKm77Rvdd4vrG1GiCw98WriMtNbX/psYzr/RikIcpEU
pm4PPXzPPFuBzYIeDFG50aPEJu6arup5b1w7SQe6lq/f/XhKYWENH1LcQOFsMoQ8oUS/WsYQ8aeT6/FxjMumjv4f9LanUHb73bB
PA0xiqtEfNIuK1ZogXgqT0157tqbmg2+GCSz+dGZv3VbSyQPdqh5s8YEGEK873vAgMBAAGjYzBhMA8GA1UdEwEB/wQFMAMBAf8w
DgYDVR0PAQH/BAQDAgEGMB0GA1UdDgQWBBRaQnES9aDCSV/XOklJczxnqxI4/DAfBgNVHSMEGDAWgBRaQnES9aDCSV/XOklJczx
nqxI4/DANBgkqhkiG9w0BAQsFAAOCAQEAHyxu0Z74ZYbWdy/fUtI9uIid/7F5AjbDdTzJcZgbSvyF3ZYVF62pRjSyxtIcCKbbr/
oRPf5voYzlIP2UL7HGBB1WzKDnfP5sXWWEC5kYmo7NrYxTzbg22mS7nUpiro07qr1FTM1aCaJhu1RqioUKX4omlilZqTkTq6lBm
DOdN+5RyBoA28EV+stt3+NV1JzOxIFqEqJfMW1q4Bzg5RM/S4xy/jCj/hMSn2Etm5YoNVwju2L86JZ8433SoemQWjl7qMHEJ1tT
MEG9hR5DiE9j6STtbza+WbJHGqSdY/z1IsYbNgoZgYtJbRtZ4aObZb4FxflMTvObXiOInYgeKdK+Dw=="
                      ]
                  }
              },
            "SignatureValue":
"MEYCIQCCAxLBoPw5h8hW4MQJhFM6K6Y8lN8JXb3Vci4N4P27YwIhAMro6KQGXRbw3x1C5h6L0Qp0L5t0XscOTPWXE67c1SCT"
          },
        "Additional": "Not signed since it comes after the EnvelopedSignature"
      }
  }
```

Canonicalization

The principle for canonicalization is simple: All textual data between the JSON object pointed out by the `Reference` down to and *including* the `SignatureInfo` element is used after *removing all whitespace*. Also worth noting is that this scheme (unlike most other JSON-based systems), depends on that *strict property order is honored*.

The sample signature has the following canonicalization data:

```
"TestSignatures":{"@jmns":"http://example.com/signature","Now":"2013-08-30T07:56:08+02:00","HRT":{"
RTl":"67","YT":{"HTL":"656756#","INTEGER":-689,"Fantastic":false},"er":"33"},"ARR":[],"BARR":[{"HTL
":"656756#","INTEGER":-689,"Fantastic":true},{"HTL":"656756#","INTEGER":-689,"Fantastic":false}],"I
D":"lADU_sO067Wlgoo52-9L","STRINGS":["One","Two","Three"],"EscapeMe":"A\\\n\"","Intra":78,"Envelope
dSignature":{"SignatureInfo":{"Algorithm":"http://www.w3.org/2001/04/xmldsig-more#ecdsa-sha256","Re
ference":{"Name":"ID","Value":"lADU_sO067Wlgoo52-9L"},"KeyInfo":{"SignatureCertificate":{"Issuer":"
CN=Demo Sub CA,DC=webpki,DC=org","SerialNumber":1377713637130,"Subject":"CN=example.com,O=Example O
rganization,C=US"},"X509CertificatePath":["MIIClzCCAX+gAwIBAgIGAUDGIccKMA0GCSqGSIb3DQEBCwUAMEMxEzAR
BgoJkiaJk/IsZAEZFgNvcmcxFjAUBgoJkiaJk/IsZAEZFgZ3ZWJwa2kxFDASBgNVBAMTC0RlbW8gU3ViIENBMB4XDTEyMDEwMTA
wMDAwMFoXDTIwMDcxMDA5NTk1OVowQjELMAkGA1UEBhMCVVMxHTAbBgNVBAoTFEV4YW1wbGUgT3JnYW5pemF0aW9uMRQwEgYDVQ
QDEwtleGFtcGxlLmNvbTBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABECkenu7FrOpy7J2FGeO1vrtseQqJT2GsaExxK5UVKe1z
hFXjF+V8OFjv/FdM9fqdqwkP/YUnx5epvvHh/+/cQWjXTBbMAkGA1UdEwQCMAAwDgYDVR0PAQH/BAQDAgP4MB0GA1UdDgQWBBR4
YF2UOnLWDhOPLLuXYZum7xLMajAfBgNVHSMEGDAWgBRZXCF2vVvvaakHecbUVh7jS1yIVTANBgkqhkiG9w0BAQsFAAOCAQEAjBu
ZK2TcDhib12DSW8rW3kyjfQ3iYtjNSVd7vJ5jyI+0OYQ/NlhN4vVJx7Z02vnrBxv1Nh9swgT5Jpw0724KawGC4P+/bUEvKVz89t
PMl9DaV98yQ2YN4cBfhcW3FpAoI4dzBbCzfEplsh9Ek7VxuIgwPozl0AdqOmTjZ3hh54ApSq/PMwENDyCEzD6bvrCrqCjgWSYIQ
UIvQ7LfO2HAlEE9DcoV4mSl/8uiQ05hRdGmNYUHZVUua0HHX1h/nAS+IcS6/EDd89kEGrL3M92a5wqnIQvDLO2NBCXhHSxoPVyB
zv0lIgaO0ixD+q5P2OszRBYG3uk9W/uNIHdoyQn19w==","MIIDZjCCAk6gAwIBAgICAMgwDQYJKoZIhvcNAQELBQAwRDETMBEG
CgmSJomT8ixkARkWA29yZzEWMBQGCgmSJomT8ixkARkWBndlYnBraTEVMBMGA1UEAxMMRGVtbyBSb290IENBMB4XDTA1MDcxMDE
wMDAwMFoXDTI1MDcxMDA5NTk1OVowQzETMBEGCgmSJomT8ixkARkWA29yZzEWMBQGCgmSJomT8ixkARkWBndlYnBraTEUMBIGA1
UEAxMLRGVtbyBTdWIgQ0EwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCQI1w6lq0AsHbWMes8i/UGBeVQnlbzL2N8V
yLjkbT3HXNHPUTjWWQElhoRzFA2xPaH++V/ecgr2Dkievrm+B5yIsdAL4oWWmgZ9KVMSfOrl5jy843p6AA55CHlP4j8v1uU1Spe
xIMUegDcNPlBwSRc0PPX+uqQ1STRg0kUgi4Bap7U5IRxTvp06adFXU4Bjr85ML7VZ3j+164t6mLnwF5RChJMlO7aVuz6TwxnWqe
ZytjFOei742dgbX9SHPVvytLtbFp4V/VFoEhaOXLZiOudPvpVwVdlfgE0AtiGHEWrfA74BU5XhME6UXzjcl3y3Ic304YGymo2jv
mOwBki5wb3AgMBAAGjYzBhMA8GA1UdEwEB/wQFMAMBAf8wDgYDVR0PAQH/BAQDAgEGMB0GA1UdDgQWBBRZXCF2vVvvaakHecbUV
h7jS1yIVTAfBgNVHSMEGDAWgBRaQnES9aDCSV/XOklJczxnqxI4/DANBgkqhkiG9w0BAQsFAAOCAQEAMlPdBaZ/+AMDfFYI9SLQ
enx0/vludp0oN9BSDe+mTfYNp5nS131cZRCKMAR3g/zzgkULu022xTJVsXfM1dsMYwEpGZp+GAvrlmRO6IathHW4aeo0QpaygOg
fquQNYgS3Z8OJRSUDGnoY65g50dgvl1+ASbZX/r0/fNANLzXt/cnf0VXPrWdqvhuUSO561TsbTYg4qzcyDRV5vpjoUAxjFna06T
JkeZR/OYMMcTtPRJON3/bMvzp7MFoL20PRPxu8nnqxwLWNzoQCkExS2yWHq1YDNNL4C/PIuyC/2IUbbPuwNp8ir3MVDBq4QwuXb
w6xFvbPsxOmZyH10xvpsnmokg==","MIIDZjCCAk6gAwIBAgIBATANBgkqhkiG9w0BAQsFADBEMRMwEQYKCZImiZPyLGQBGRYDb
3JnMRYwFAYKCZImiZPyLGQBGRYGd2VicGtpMRUwEwYDVQQDEwxEZW1vIFJvb3QgQ0EwHhcNMDIwNzEwMTAwMDAwWhcNMzAwNzEw
MDk1OTU5WjBEMRMwEQYKCZImiZPyLGQBGRYDb3JnMRYwFAYKCZImiZPyLGQBGRYGd2VicGtpMRUwEwYDVQQDEwxEZW1vIFJvb3Q
gQ0EwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCMR4ROlJJyBUzQ92XDSzFuxiMjwFqsrKXuIksIXMypjg2QZF4Pzy
Q0pu32/LVKuoaqbT+bkRKFdpUvMKhzGQ3rMAthTUkhXpFJN5Bk2LTcGXoE0B9wPKn4C3cxbUMEtT94m8PKIjRoKm77Rvdd4vrG1
GiCw98WriMtNbX/psYzr/RikIcpEUpm4PPXzPPFuBzYIeDFG50aPEJu6arup5b1w7SQe6lq/f/XhKYWENH1LcQOFsMoQ8oUS/Ws
YQ8aeT6/FxjMumjv4f9LanUHb73bBPA0xiqtEfNIuK1ZogXgqT0157tqbmg2+GCSz+dGZv3VbSyQPdqh5s8YEGEK873vAgMBAAG
jYzBhMA8GA1UdEwEB/wQFMAMBAf8wDgYDVR0PAQH/BAQDAgEGMB0GA1UdDgQWBBRaQnES9aDCSV/XOklJczxnqxI4/DAfBgNVHS
MEGDAWgBRaQnES9aDCSV/XOklJczxnqxI4/DANBgkqhkiG9w0BAQsFAAOCAQEAHyxu0Z74ZYbWdy/fUtI9uIid/7F5AjbDdTzJc
ZgbSvyF3ZYVF62pRjSyxtIcCKbbr/oRPf5voYzlIP2UL7HGBB1WzKDnfP5sXWWEC5kYmo7NrYxTzbg22mS7nUpiro07qr1FTM1a
CaJhu1RqioUKX4omlilZqTkTq6lBmDOdN+5RyBoA28EV+stt3+NV1JzOxIFqEqJfMW1q4Bzg5RM/S4xy/jCj/hMSn2Etm5YoNVw
ju2L86JZ8433SoemQWjl7qMHEJ1tTMEG9hR5DiE9j6STtbza+WbJHGqSdY/z1IsYbNgoZgYtJbRtZ4aObZb4FxflMTvObXiOInY
geKdK+Dw=="]}}
```

Other JSON Signature Solutions

The IETF JOSE WG has defined a JSON signature scheme called JWS. The primary reason why I haven't adopted JWS for KeyGen2 is because it is based on *in-line signatures using Base64-encoded payloads*.

Although certainly working Base64-encoded messages disrupts readability making the switch from XML to JSON unnecessary painful for schemes where the *message* is the core and a signature only is there to vouch for the message's authenticity. The following shows how a JWS-based conversion of the sample message could look like:

```
{
  "payload": "dTzJcZgb ... QWBBRaQnES",
  "signatures":
    [{
          Signature data
    }]
 }
```

Yet another scheme which is quite similar to this specification is something known as "HTTP Keys":

https://payswarm.com/specs/source/http-keys

The authors of HTTP Keys also created their own signature scheme for multiple reasons, with clear-text messaging as one objective.

Author: Anders Rundgren

anders.rundgren.net@gmail.com

2013-08-30