Signature Validation Token

Security Dispatch IETF 107

Stefan Santesson

stefan@aaa-sec.com

Goal

Simple solution for validating signatures in a distant future

The time machine approach

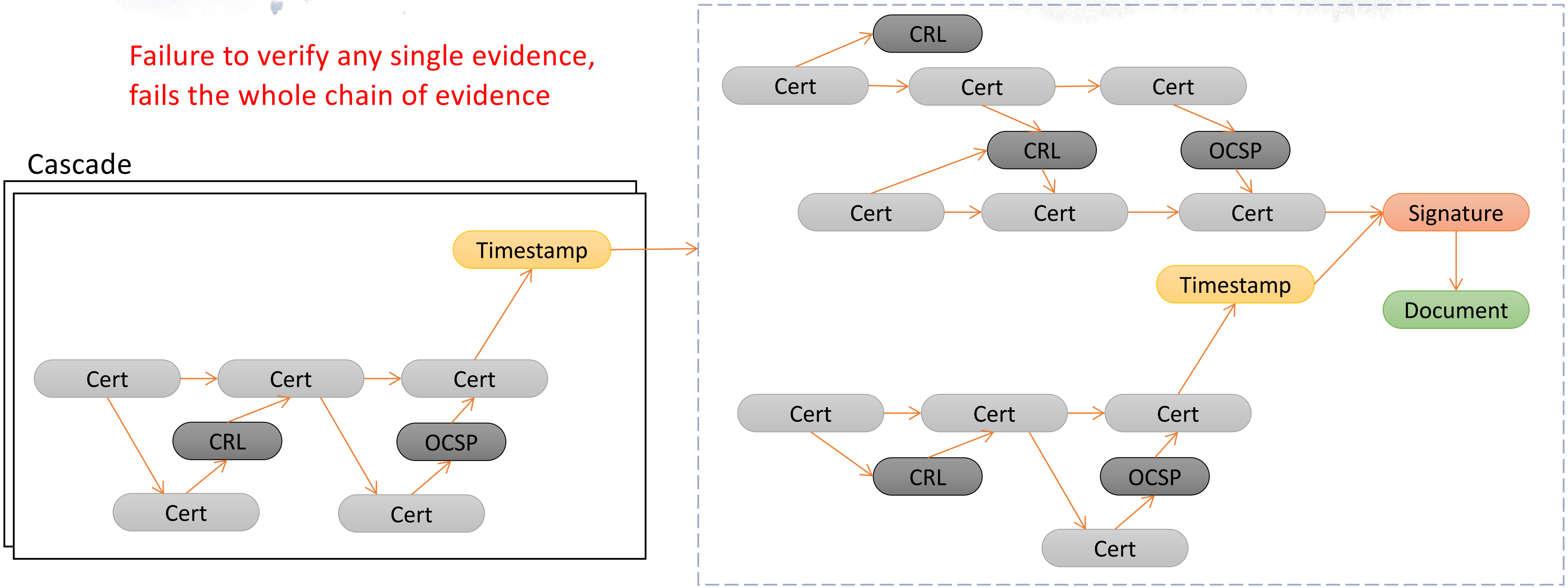
Bring the verifier to a point in time when the signature was fresh, certificates were trusted, and algorithms were secure

Time machine

- Establish a time when the signature existed
- Prove that the certificates were valid at the time
- Prove that the signed document matching this signature existed when the signature algorithm was still considered secure

Time machine – comlex chains of evidence

Failure to verify any single evidence,
fails the whole chain of evidence



Time machine problems

- Very complex – Number of signed objects can be in the range of 50 – 100 in extreme cases, each necessary to prove validity
- Current standards are incomplete (revocation info supporting time stamps not mandatory)
- Eventually, each signature will be impossible to validate with certainty at some point in time

We need a new paradigm

SVT
A simple counter proposal to the time machine

01

SVT is validated in current time (No more time machine)

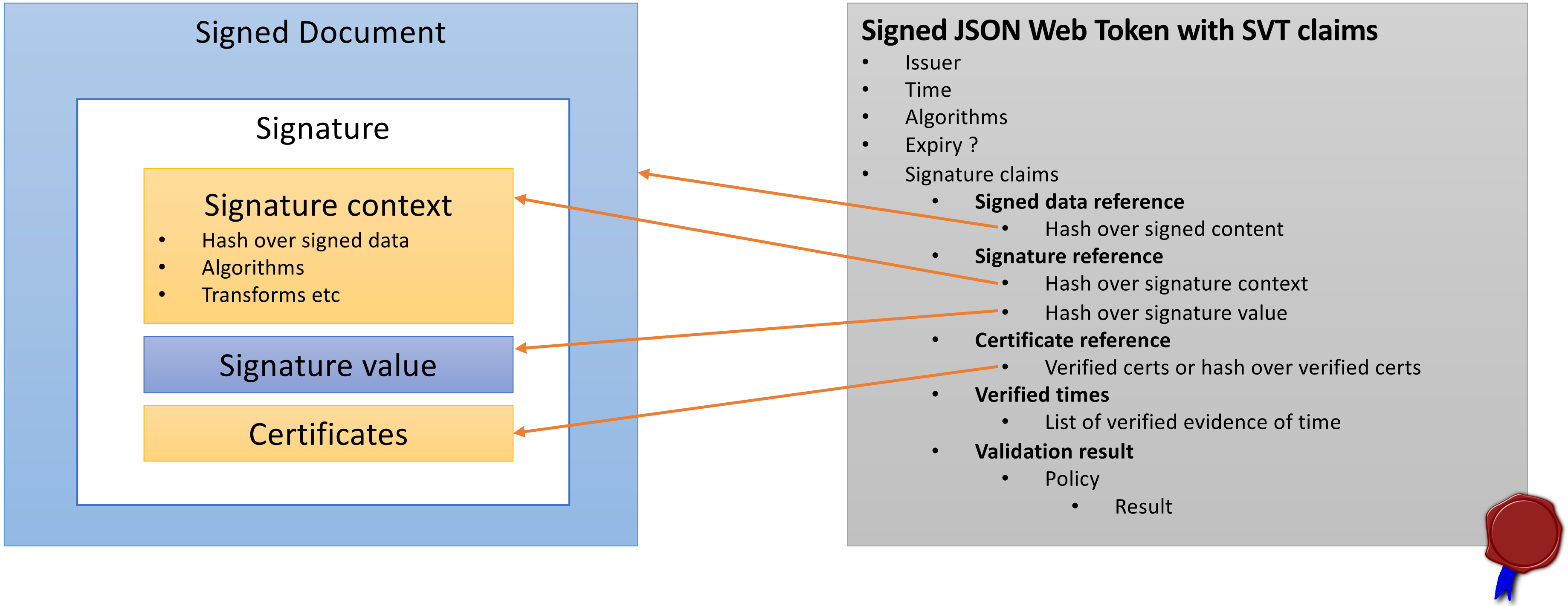
02

Removes the need to validate the original signature and to use any of the original algorithms

03

All achieved by 1 signed statement, signed by 1 currently trusted key using 1 curenlty trusted algorithm

SVT – How it works



Signed Document

Signature

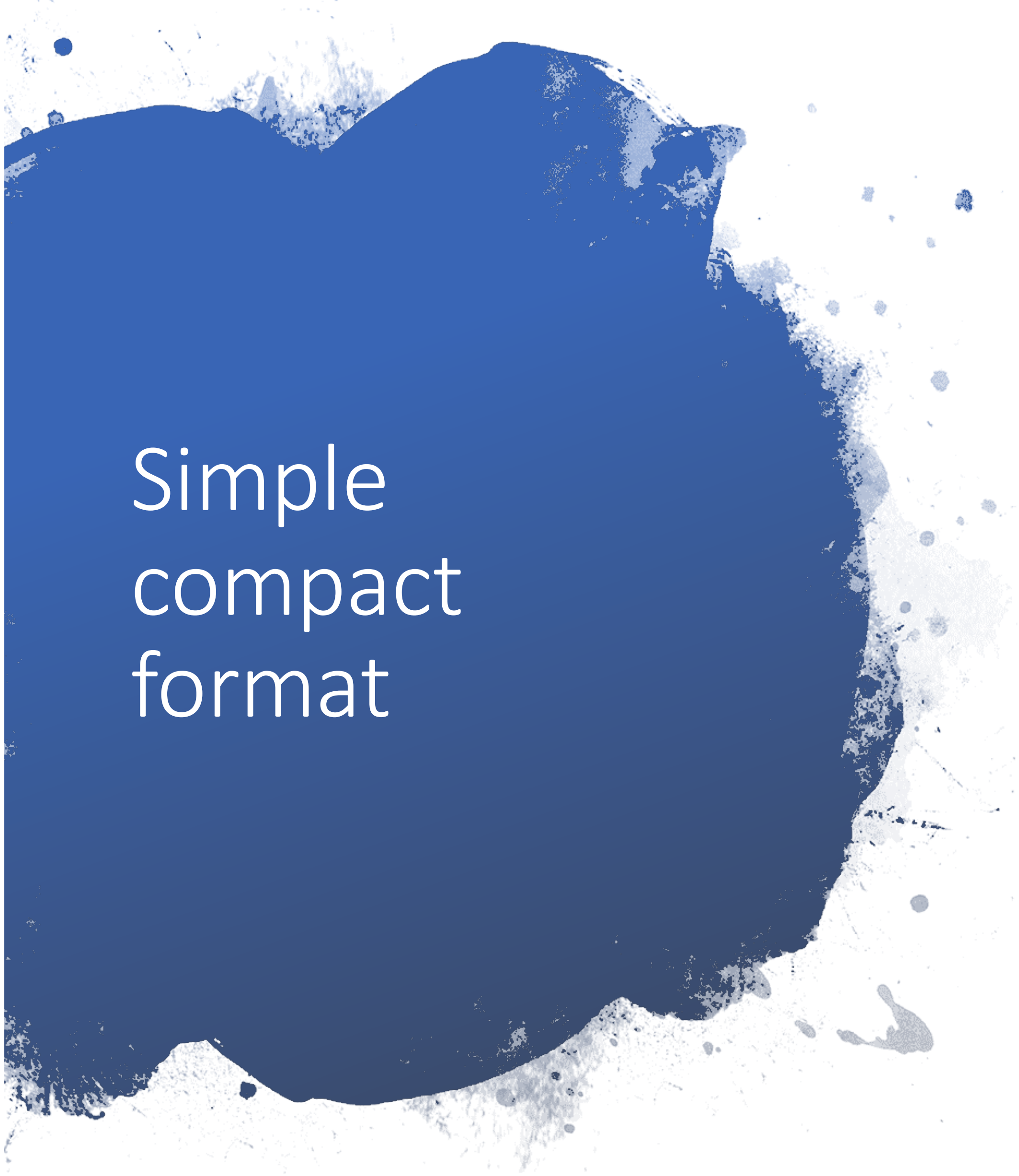
Signature context
- Hash over signed data
- Algorithms
- Transforms etc

Signature value

Certificates

Signed JSON Web Token with SVT claims

- Issuer
- Time
- Algorithms
- Expiry ?
- Signature claims
  - **Signed data reference**
    - Hash over signed content
  - **Signature reference**
    - Hash over signature context
    - Hash over signature value
  - **Certificate reference**
    - Verified certs or hash over verified certs
  - **Verified times**
    - List of verified evidence of time
  - **Validation result**
    - Policy
      - Result

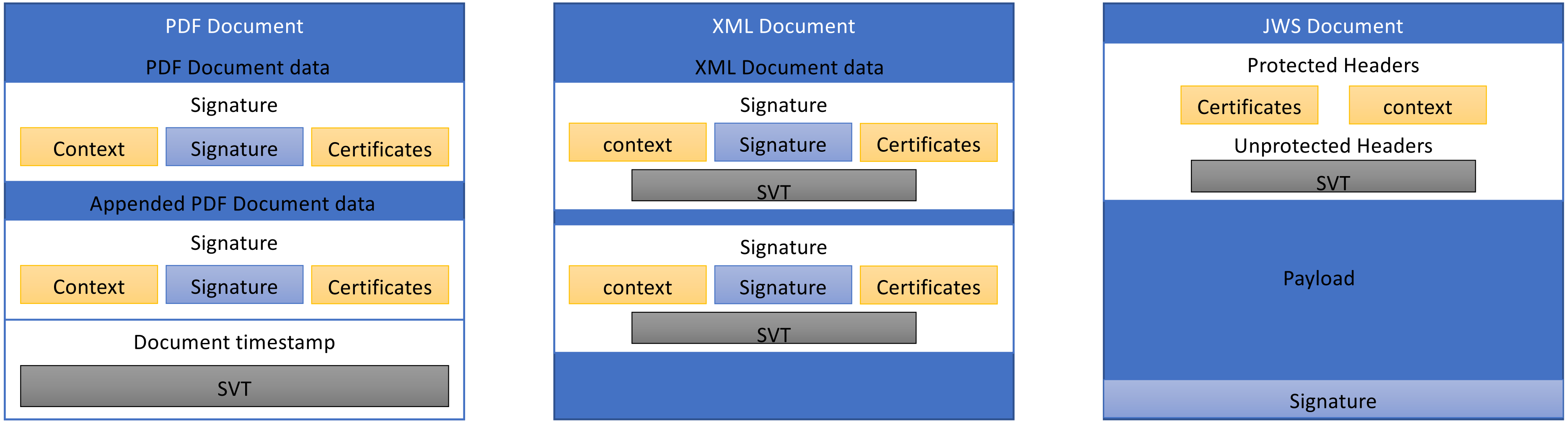


Simple compact format

SVT JWT Calims

```
{
 "aud" : "http://example.com/audience1",
 "iss" : "https://swedenconnect.se/validator",
 "iat" : 1584703056,
 "jti" : "45d4f765d1f981f7f0c304615ad9491",
 "sig_val_claims" : {
   "sig" : [ {
     "sig_val" : [ {
       "msg" : "Passed basic signature validation",
       "res" : "PASSED",
       "pol" : "http://id.swedenconnect.se/svt/sigval-policy/chain/01"
     } ],
     "sig_ref" : {
       "sig_hash" : "mC0ReA...Vqdw==",
       "sb_hash" : "DNn...aXg=="
     },
     "signer_cert_ref" : {
       "ref" : [ "fIdr...UnoA==" ],
       "type" : "chain_hash"
     },
     "sig_data_ref" : [ {
       "ref" : "0 74697 79699 37821",
       "hash" : "qmIjbB...5ihujvw=="
     } ],
     "time_val" : [ ]
   } ],
   "ver" : "1.0",
   "profile" : "PDF",
   "hash_algo" : "http://www.w3.org/2001/04/xmlenc#sha512"
 }
}
```

Implementation profiles for PDF, XML, JWS, …



Signature Validation Token

What is claims:

- Trust service (A) performed the validation process (B) to this signature with the following result (C)!

Because this statement never changes

What it does NOT claim:

- This signature is valid!

Because this statement may change

Some criticism so far

How can I trust the SVT issuer?	All signature validation depends on statements by TTP:s. The assumption is that is better to trust one statement rather than 50+ statements, where all must be accurate.
What if the SVT gets to old to verify in a current time context?	A new SVT can be issued based on an old SVT, carrying over the original statements. Other suitable techniques like block chaining can also be used to guarantee authenticity of original SVT.
It obviously better to do my own verification of the original signature, than trusting an old statement from a validation service.	Not if we look closely. All signature validation requires the verifier to trust statements of validity that must be accurate, such as certificates, revocation data, timestamps etc. The time machine doesn’t perform better here, but worse.

Some criticism so far

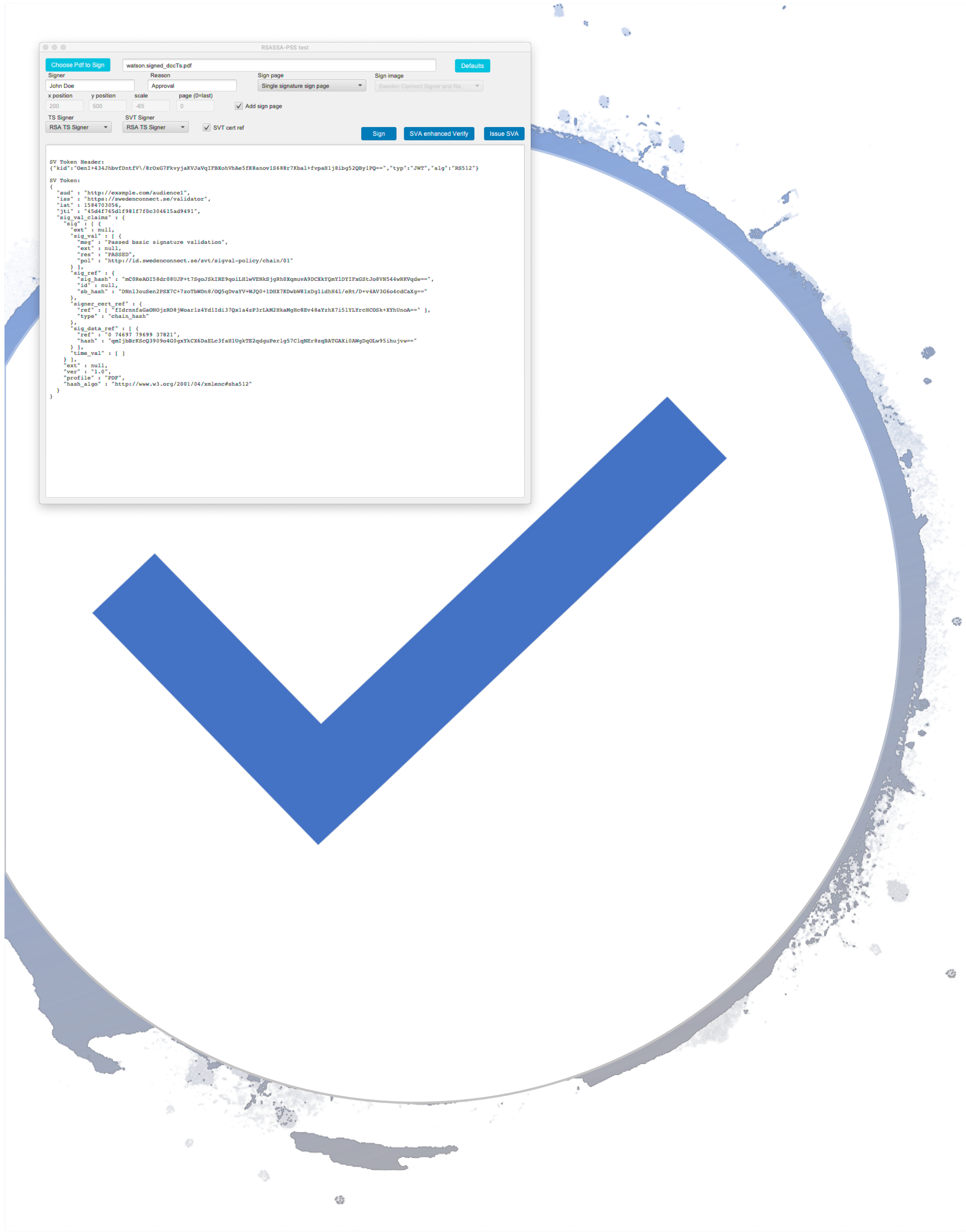
Why would I trust your validation service?

The basic idea is the the SVT is added by the relying party on first verification and before archival. The verifier will then pick an SVT issuer that is relevant to the risks involved.

Will I have to use special tools to view the signed document?

The SVT is added to each document format in a way that is ignored by a standard tool. E.g. The PDF SVT appears like a timestamp, the XML SVT appear as additional Object data.

Status



watson.signed_docTs.pdf

Signer: John Doe

Reason: Approval

Sign page: Single signature sign page

Sign image: Sweden Connect Signer and Re...

x position: 200

y position: 500

scale: -65

page (0=last): 0

☑ Add sign page

TS Signer: RSA TS Signer

SVT Signer: RSA TS Signer

☑ SVT cert ref

```
SV Token Header:
{"kid":"OenI+434JhbvfDntfV\/8rOxG7FkvyjaKVJaVqIFBXohVhAe5fK8anov1S688r7Kbal+fvpaH1j8ibg52QBy1PQ==","typ":"JWT","alg":"RS512"}

SV Token:
{
  "aud" : "http://example.com/audience1",
  "iss" : "https://swedenconnect.se/validator",
  "iat" : 1584703056,
  "jti" : "45d4f765d1f981f7f0c304615ad9491",
  "sig_val_claims" : {
    "sig" : [ {
      "ext" : null,
      "sig_val" : [ {
        "msg" : "Passed basic signature validation",
        "ext" : null,
        "res" : "PASSED",
        "pol" : "http://id.swedenconnect.se/svt/sigval-policy/chain/01"
      } ],
      "sig_ref" : {
        "sig_hash" : "mC0ReAOI58dr08UJP+t7SgoJSkIRE9qoiLHlwVENkSjgRh0XqmuvA9DCXkYQmYlDYIFxGStJo8VN544wRKVqdw==",
        "id" : null,
        "sb_hash" : "DNnl3ouSen2PSX7C+7zoTbWDn8/OQ5qDvaYV+WJQ0+1DHX7KDwbW81xDglidhH4l/eRt/D+v4AV3G6o4cdCaXg=="
      },
      "signer_cert_ref" : {
        "ref" : [ "fIdrnnfaGaONOjzRD8jWoarlz4Yd1Idi37Qx1a4zP3rLkM2HkaMgHc8Ev48aYzhX7i5lYLYrcHCOSk+XYhUnoA==" ],
        "type" : "chain_hash"
      },
      "sig_data_ref" : [ {
        "ref" : "0 74697 79699 37821",
        "hash" : "qmIjbBrKScQ3909o4G0gxYkCX6DaELc3faH1UgkTE2qdguPerlg57ClqNEr8zqBATGAXi0AWgDqOLw95ihujvw=="
      } ],
      "time_val" : [ ]
    } ],
    "ext" : null,
    "ver" : "1.0",
    "profile" : "PDF",
    "hash_algo" : "http://www.w3.org/2001/04/xmlenc#sha512"
  }
}
```

- Government funded research project in Sweden
- Specificatioins available on GitHub: https://docs.swedenconnect.se/technical-framework/index.html#sigval
- Running code for PDF and XML in Java
- Open source implementation will be available at latest in September 2020

Why IETF?

- Based on the IETF JWT format
- Can support IETF signature formats (CMS, JWS, …)
- No other standards organization is doing this
- IETF has done similar work in the past
- It is a very important subject. Archival of signed electronic documents provide huge cost savings with greatly improved performance.
- ETSI has already built time machines (PAdES, CAdES, XAdES and soon JAdES). They are huge and complex and the only thing available. A standard from IETF would provide an alternative that could balance the scale and provide a viable alternative.

Where in the IETF?

LAMPS?

Questions