

Security Token Service (STS)



Henri Mikkonen

Helsinki Institute of Physics

4th EMI All-Hands Meeting

8.-10.5.2012

Hamburg, Germany

EMI is partially funded by the European Commission under Grant Agreement RI-261611

- Terminology
- Functionality
- (Some) STS Use Cases
- Server-side Architecture
- Current state of the implementation
- Requirements for the clients
- Discussion

- Security Token?

- « A collection of statements (claims) about a user or a resource »

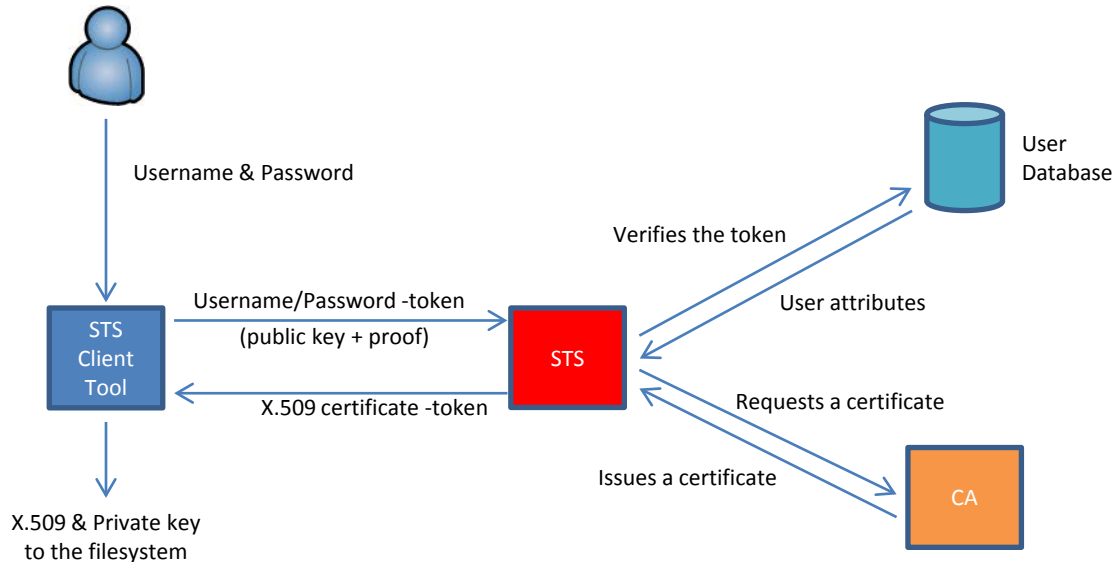
- *Anything that can be attached into a Web Service (SOAP) message*
 - *Example token formats: X.509 certificate, SAML assertion, Kerberos ticket, Username/Password, ...*

- Security Token Service?

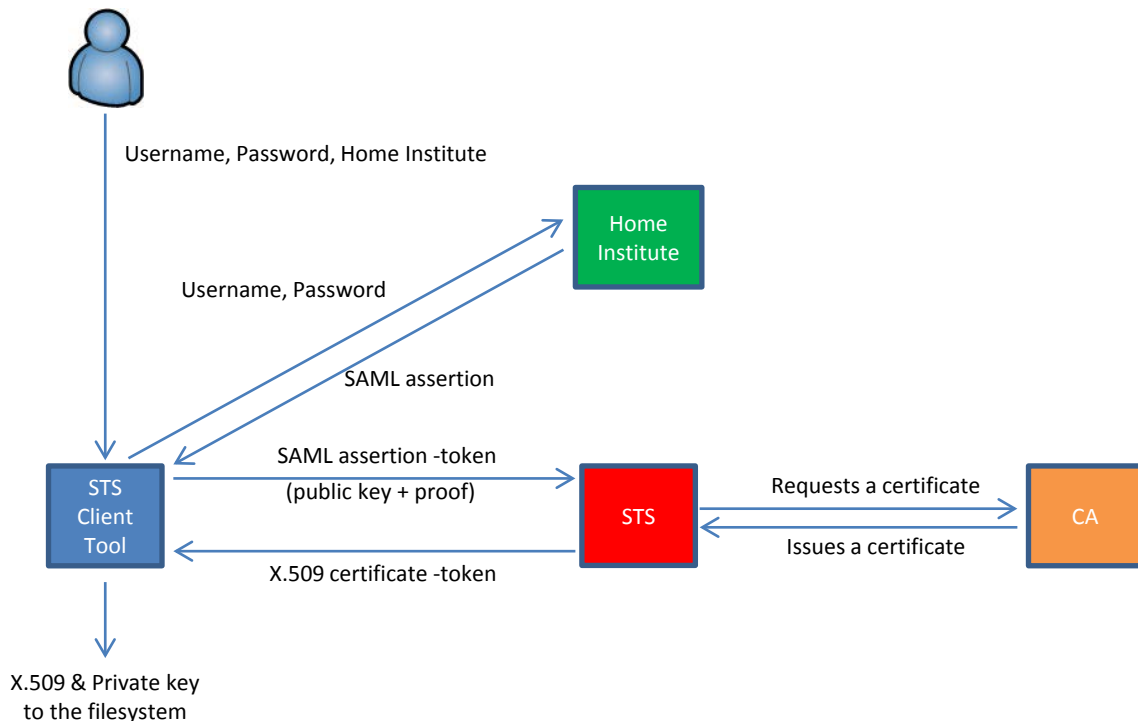
- « A Web service used to issue, renew, validate and cancel security tokens »

- Transforms the security token into another security token
 1. Validates the incoming security token
 2. Aggregates the required information & issues the new security token
 - *Possibly by exploiting external sources / authorities*
 3. Includes the new security token into the response message
- Establishes a trust relationship between different application domains

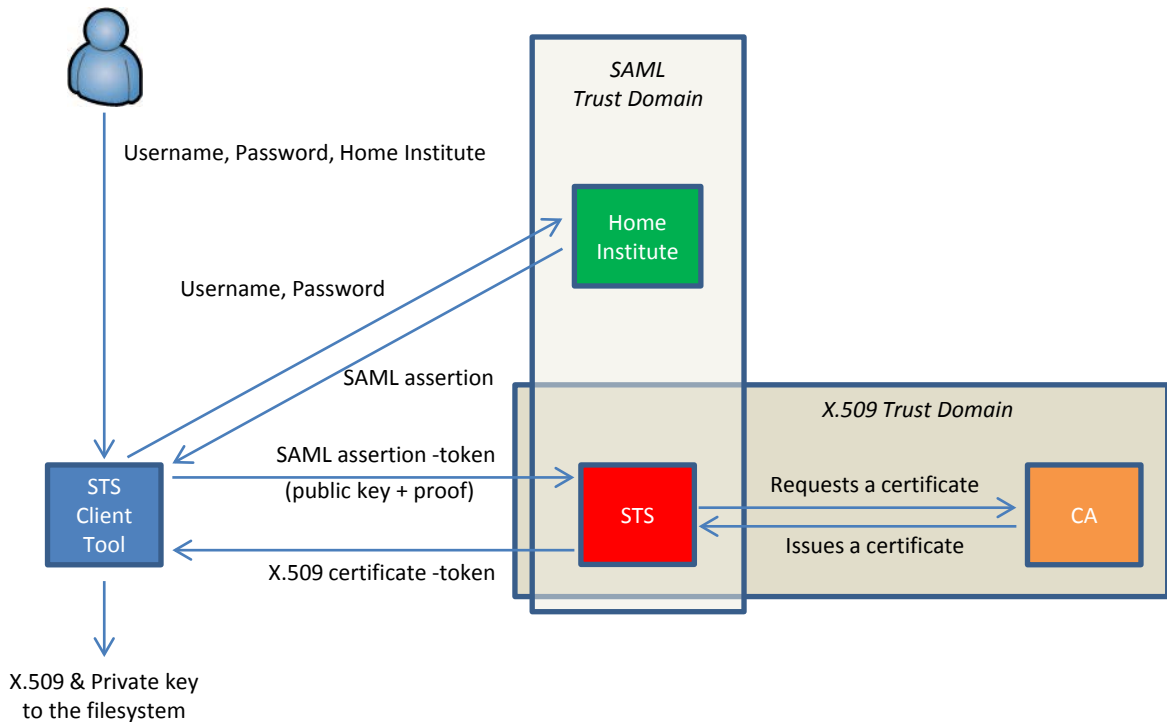
Username/Password to X.509



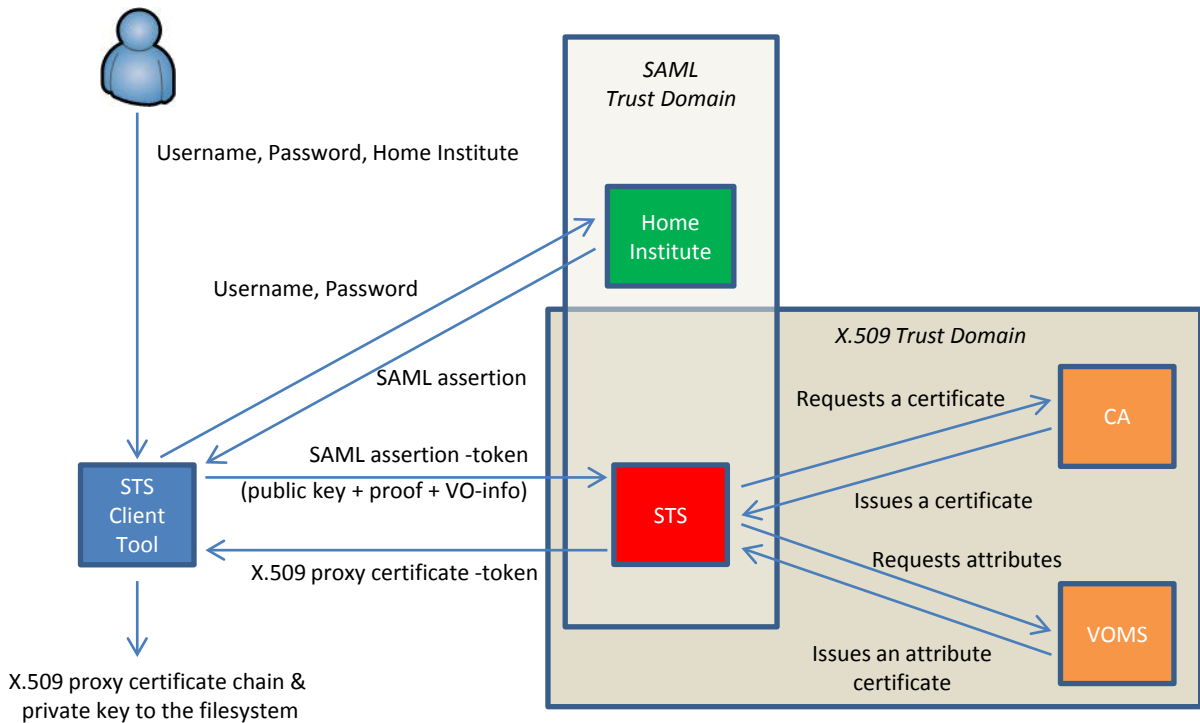
SAML assertion to X.509



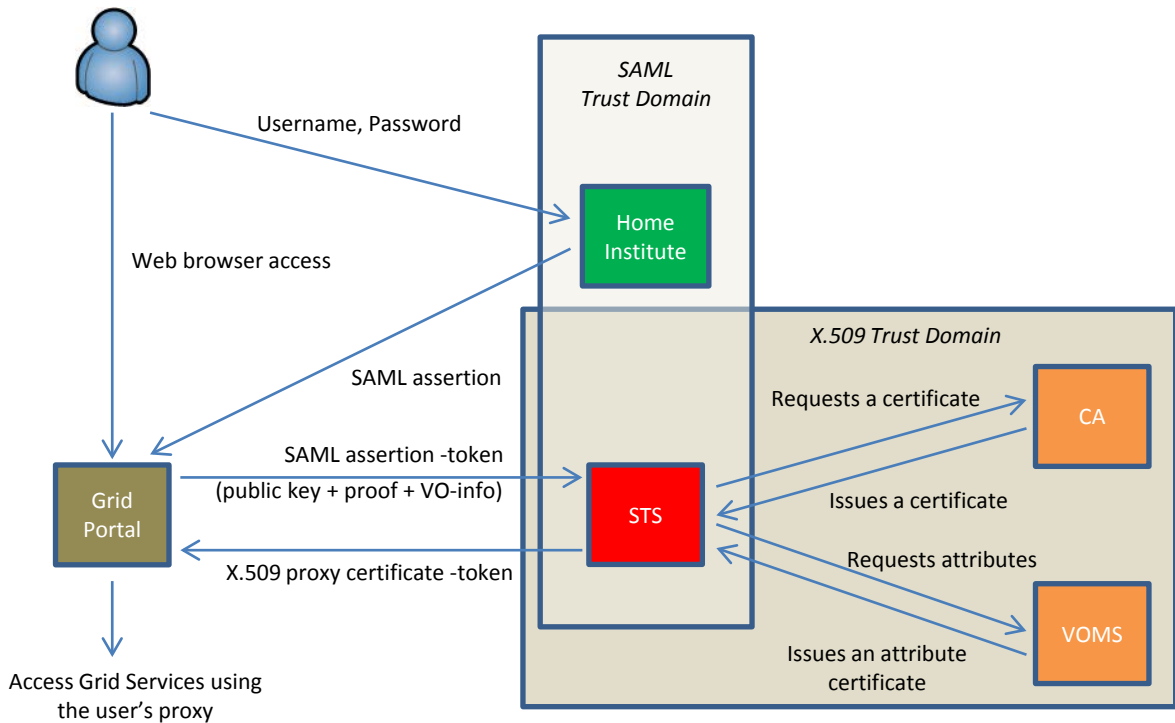
SAML assertion to X.509



SAML assertion to X.509 proxy

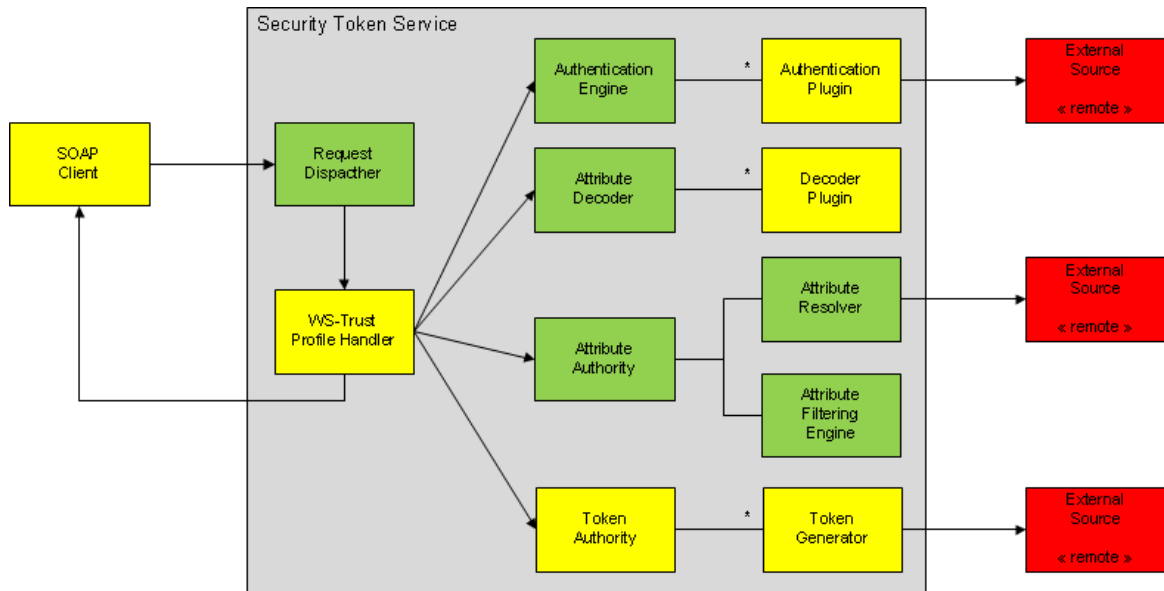


SAML assertion to X.509 proxy

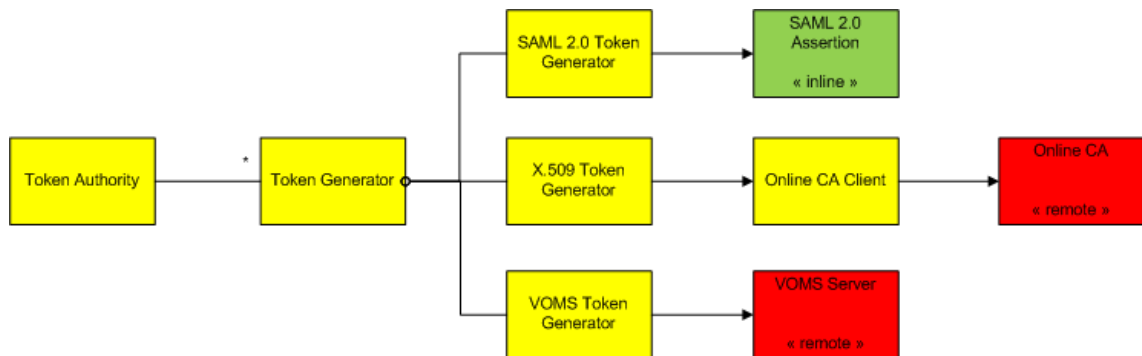


- STS transforms an existing security token into another security token
 - Supported incoming formats: X.509, X.509 proxy, Username/Password, SAML, Kerberos
 - Supported outgoing formats: X.509, X.509 proxy, SAML
- STS is a SOAP-based Web Service
 - Any party capable of producing specified request messages and understanding response messages can act as a client
 - *Command-line clients, Web portals, Grid resources, ...*

Server-side Architecture (1/2)



OpenSAML3 is used for producing and consuming the protocol messages
The *green* components are / will be provided by Shibboleth 3 Identity Provider
The *yellow* components are currently being implemented in the STS project



- Three outgoing token formats promised in the EMI plans
 - SAML 2.0 assertion generation is provided “internally” by Shib IdP
 - The CMP protocol is initially supported for the Online CA connection
 - The VOMS token generation is very close to *voms-proxy-init* functionality
 - Some building blocks already seem to exist for Java

- OpenSAML3 and Shib IdP v3 starts to have most of the required functionality implemented
- X.509 token generator using the CMP protocol is ready
 - Communication with EJBCA is tested to be working
 - Others (e.g. MyProxy) may be supported

- Obtain the data needed for creating an incoming security token
 - Username/Password: simple, prompt the user
 - X.509: potentially proof of private key possession
 - SAML Assertion: it must be targeted to the STS
 - *Building blocks: SAML delegation for SPs and the ECP profile for client tools*
- Encode the security token
 - WS-Security namespace

- Generate the request message
 - SOAP envelope, WS-Security token in the header, WS-Trust RST-message in the body
 - *Optionally data related to the security token to be issued*
 - Eg. Public key to an X.509 certificate or a list of VO roles desired to be included in a VOMS proxy
 - *Optionally XML-Signature and/or XML-Encryption*
- Send the request message to the STS
 - https server-side authentication by default

- Validate the response message
 - SOAP envelope, WS-Security in the header, WS-Trust message in the body
 - Potentially XML-Signature and/or XML-Encryption
- Decode the new security token
 - Eg. X.509 token is Base64-encoded
 - Potentially combine with some earlier-generated data
 - *Eg. the corresponding private key*

- Can be relatively straight-forward..
 - Eg. Username/Password -> X.509 certificate, using a single User database/directory
- ..but potentially quite challenging
 - Eg. Using SAML2 assertion as an incoming token, supporting XML-Signature & XML-Encryption
 - *Obtaining the SAML assertion have some, potentially additional requirements for the assertion issuers*
 - *XML-Signature & XML-Encryption are not simple*

Thank you! Questions?

Henri Mikkonen

<henri.mikkonen@cern.ch>