Trust model for EMAM

how can it help our project

Victoriano Giralt proxying Milan Sova

CESNET
Central ICT Services University of Málaga

RS^{3}G Mobility Coding Camp
Barcelona
March 24th 2011

Outline

Context

starting point

How do we want security in our system

- Features we need
  - Integrity
  - Confidentiality
  - Authenticity
- Features we do not need
  - Non-repudiation
- Security provided by transport
  - Transport Layer Security (*TLS*)

Integrity

part of the channel

TLS provides integrity on the transmission channel

- Recipient will notice message modifications in transit
- No special measures required

SSEDIC Programme Partner

Confidentiality

part of the channel

TLS provides confidentiality on the transmission channel

- Transport is encrypted
- Only end-points can see message content
- No special measures required

Authenticity

channel setup also helps

The TLS certificates can be put to use here

- Peers can be identified by way of their TLS certificates
- Peer authorisation can be based on TLS certificates
- Number of accepted certificate issuing authorities (CAs) should be limited

SSEDIC Programme Partner

Authorisation

TLS to the rescue again

EMAM node TLS certificates can

- Identify the node by the certificate Subject
- The registry can provide extra information
  - Node owner organisation
  - Node acceptable roles
    - Consumer
    - Provider
  - ...

Common configuration

security setup for all nodes

TLS stack configuration

- Use and accept only strong cipher suites
  - TLS_RSA_WITH_AES_256_CBC_SHA
  - TLS_RSA_WITH_AES_256_CBC_SHA256
- Clients: accept only peer certificates from trusted CAs
- Servers: request only client certificates from trusted CAs
- Use certificates from a trusted CA
- Trusted CAs:
  - TERENA Certificate Service (TCS)

Trust anchor

feeding the starter pump

Every node stores the certificate Subject(s) of the EMAM registry node(s) in local configuration

TLS Server Authorisation by Client

Steps to start a conversation, or not

1. get *ServerURL* and server certificate subject (*SSubjectExpected*) from EMAM registry
2. connect to the Server using TLS
3. extract the server certificate from TLS stack
4. extract subject (*SSubject*) from the certificate
5. SSubject $=$ SSubjectExpected ?
   - $=$ Authorise
   - $\neq$ Reject

TLS Server Authorisation by Client

in pseudocode

```
(SServerExpected, ServerURI) = readRegistry(org);
ServerCert = TLSconnect(ServerURI);
SSubject = getSubject(ServerCert);
if (not(DNequal(SSubject, SSubjectExpected))
  return NOT_AUTHORIZED;
```

TLS Client Authorisation by Server

Steps to continue a conversation, or not

1. Accept TLS connection
2. Extract client certificate from TLS stack
3. Extract the certificate subject from the certificate
4. Get client information from EMAM registry
5. Use the client information with request ACL
   is the client authorised to do what it wants?

SSEDIC Programme Partner

TLS Client Authorisation by Server

in pseudocode (variant 1)

```
connection = TLSaccept();
CCert = getCertificate(connection);
CSubject = getSubject(CCert);
request = getRequest(connection);
COrg = readReg(CSubject);
if (not(requestAllowed(request, COrg))
  return NOT_AUTHORIZED;
```

TLS Client Authorisation by Server

in pseudocode (variant 2)

```
connection = TLSaccept();
CCert = getCertificate(connection);
CSubject = getSubject(CCert);
request = getRequest(connection);
COrg = getOrg(request);
// is the client registered to act
// on behalf of the organization?
match = checkRegistry(COrg, CSubject);
if (not(match))  return NOT_AUTHORIZED;
if (not(requestAllowed(request, COrg))
   return NOT_AUTHORIZED;
```

Registry authentication by Nodes

do we trust the trust anchor?

Finding partners in a secure way

1. Get the Registry certificate subject (*RegSubject*) and location from local configuration
2. Connect to the Registry using TLS
3. Extract the server certificate from TLS stack
4. Extract subject *SSubject* from the certificate
5. SSubject $=$ RegSubject ?
   - $=$ Authenticate
   - $\neq$ Reject

SSEDIC Programme Partner

Registry authentication by Nodes

in pseudocode

```
(RegSubject, RegURI) = readConfig(reg);
ServerCert = TLSconnect(RegURI);
SSubject = getSubject(ServerCert);
if (not(DNequal(SSubject, RegSubject))
  return NOT_AUTHENTICATED;
```

Thank you

SSEDIC Programme Partner

Let’s start the fun!