CIM security

Version4

Content

1 Introduction

The purpose of this document is to describe the security solution for the CIM interface for MDM.

1.1 References

Reference	Description/Link
[OMNIACIMSecurity]	OMNIA CIM Security
[RFC6749]	The OAuth 2.0 Authorization Framework https://tools.ietf.org/html/rfc6749
[RFC7519]	JSON Web Token (JWT) https://tools.ietf.org/html/rfc7519
[RFC7521]	Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants https://tools.ietf.org/html/rfc7521

1.2 Change Log

Revision	Description	Affected sections	Approved by
v1.0	Initial version	All	
v2.0	Added sub section with details for how to access OMNIA HES.	4.1	
v3.0	Changed access for MDM to be based on basic auth	3	
v4.0	Added resource parameter to token request	4.1	

1.3 Terminology

Term	Description

2 Summary

Server authentication is based on a trusted server certificate for TLS [OMNIACIMSecurity].

Client authentication is based on JWT tokens acquired from a single OAuth2 compliant [OMNIACIMSecurity] Microsoft AD domain forest.

- Domain joined services (OMNIA services) can use Windows Integrated Security for authentication towards AD FS.
- Non-domain joined services (MDM services) can use client certificates (preferred) or username/password for authentication towards AD FS.

MDM services can be deployed from e.g. Linux and are therefore not necessarily domain joined (even though it is technical feasible to do this).

3 Accessing MDM

All OMNIA clients will access an MDM server using basic authentication with username and password.

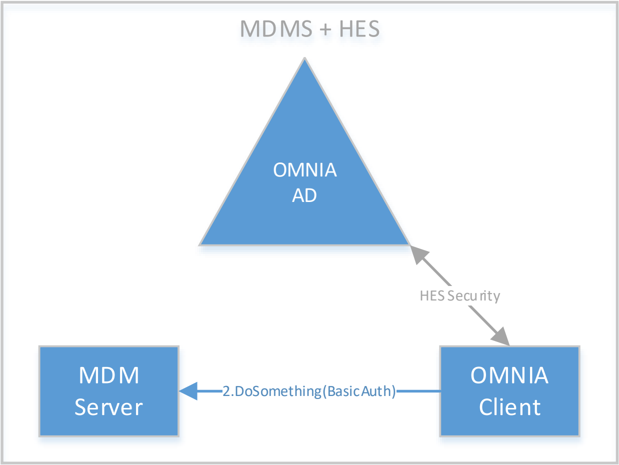

Figure 1 OMNIA client calling an MDM server

3.1 HTTP basic authentication

The Authorization header is used to send client username and password to a server.

1. The username and password are combined into a string “username:password”.
2. The resulting string literal is then encoded using the RFC2045-MIME variant of Base64.
3. The authorization method “Basic” and a trailing space is put before the encoded string

For example, if the user agent uses “Aladdin” as the username and “open sesame” as the password then the header if formed as follows:

```
Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==
```

4 Accessing OMNIA

For the MDM client to access OMNIA we will have to

1. Issue a client certificate for the MDM client.
2. Create an MDM user in AD related this user to the client certificate.

The MDM client can then

1. Acquire JWT tokens from AD using OAuth2 client assertion grant [RFC7521] based on its client certificate.
2. Access the OMNIA server using the acquired JWT token. The OMNIA server will authorize the MDM client based on the given JWT token.

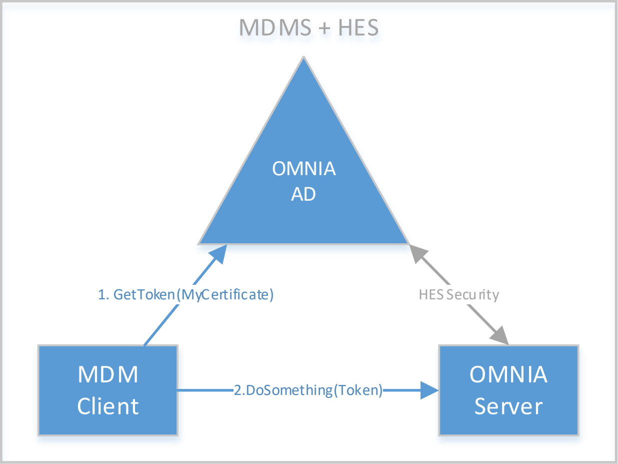

Figure 2 MDM client calling an OMNIA server

4.1 Client assertion grant based on certificate

4.1.1 Client assertion format

4.1.1.1 Example of a decoded client assertion

```
{
  "alg": "RS256",
  "x5t": "CYeXHXlZsdZDb-EJTPDoEMgNfuI",
  "typ": "JWT"
}
.
{
  "aud": "https://sts/adfs/oauth2/token",
  "exp": 1556663498,
  "iss": "bf50f2bd-19b9-497f-a575-01e8414df2f8",
  "jti": "3c6774b1-f215-452d-89c2-64916e679f6b",
  "nbf": 1556662898,
  "sub": "bf50f2bd-19b9-497f-a575-01e8414df2f8"
}
.
dr56QFMA9S9u72XwnaKEEOr0RoPKiTV79HgSs4IDmR0VzgeImqx4KRup_3gbltiKau_63IYsO1AikPL4cKB6TiT
gTUJeJQZok5IBejI5MHw9i6FR7X2btlZy4mEwVr6AJVP0XUP_2lvgRMlH4TkXkreTwaJo4OqDxToFkcS2kcrZ7T
WBhfIocfQvj5FrKS8T3s-pPvdNWiatIr-
71aXiu41Puke6H2J8NEFgrFc9w4iWZuWt9WUfLfja1RBIbU8K7JUPYkRcuowdv0xj-
lLRJHinjtD0uJex8V02QKCSGMLQ20gLKm8Ez9wlzzwzrf71gE84jaJ3IMrs7oZFB0EJmg
```

4.1.1.2 Example of the base 64 URL encoded client assertion

```
eyJhbGciOiJSUzI1NiIsIng1dCI6IkNZZVhIWGxac2RaRGItRUpUUERvRU1nTmZ1SSIsInR5cCI6IkpXVCJ9.ey
JhdWQiOiJodHRwczovL3N0cy9hZGZzL29hdXRoMi90b2tlbiIsImV4cCI6MTU1NjY2MzQ5OCwiaXNzIjoiYmY1M
GYyYmQtMTliOS00OTdmLWE1NzUtMDFlODQxNGRmMmY4IiwianRpIjoiM2M2Nzc0YjEtZjIxNS00NTJkLTg5YzIt
NjQ5MTZlNjc5ZjZiIiwibmJmIjoxNTU2NjYyODk4LCJzdWIiOiJiZjUwZjJiZC0xOWI5LTQ5N2YtYTU3NS0wMWU
4NDE0ZGYyZjgifQ.dr56QFMA9S9u72XwnaKEEOr0RoPKiTV79HgSs4IDmR0VzgeImqx4KRup_3gbltiKau_63IY
sO1AikPL4cKB6TiTgTUJeJQZok5IBejI5MHw9i6FR7X2btlZy4mEwVr6AJVP0XUP_2lvgRMlH4TkXkreTwaJo4O
```

```
qDxToFkcS2kcrZ7TWBhfIocfQvj5FrKS8T3s-pPvdNWiatIr-
71aXiu41Puke6H2J8NEFgrFc9w4iWZuWt9WUfLfja1RBIbU8K7JUPYkRcuowdv0xj-
lLRJHinjtD0uJex8V02QKCSGMLQ20gLKm8Ez9wlzzwzrf71gE84jaJ3IMrs7oZFB0EJmg
```

4.1.1.3 Header parameters

Parameter	Description
alg	Must be RS256
typ	Must be JWT
x5t	Must be the base 64 URL encoding of the X.509 Certificate SHA-1 thumbprint

4.1.1.4 Payload parameters

Parameter	Description
aud	Audience: The recipient that the JWT is intended for. That is the AD FS endpoint. Example: https://sts/adfs/oauth2/token
exp	Expiration date: The date when the token expires. The time is represented as the number of seconds from January 1, 1970 (1970-01-01T0:0:0Z) UTC until the time the token validity expires.
iss	Issuer: Must be the client_id assigned to you
jti	GUID: The JWT ID
nbf	Not Before: The date before which the token cannot be used. The time is represented as the number of seconds from January 1, 1970 (1970-01-01T0:0:0Z) UTC until the time the token was issued.
sub	Subject: As for iss, must be the client_id assigned to you

4.1.1.5 Signature

The signature, marked with green in the examples, is computed from the header and payload by applying the certificate as described in [RFC7519].

4.1.2 Request

4.1.2.1 Example of token request

```
POST /adfs/oauth2/token
Host: https://sts
Content-Type: application/x-www-form-urlencoded


client_id=bf50f2bd-19b9-497f-a575-01e8414df2f8&
client_assertion_type=urn%3Aietf%3Aparams%3Aoauth%3A
client-assertion-type%3Ajwt-bearer&
client_assertion=eyJhbGciOiJSUzI1NiIsIng1dCI6IkNZZVhIWGxac2RaRGItRUpUUERvRU1nTmZ1SSIsIn
R5cCI6IkpXVCJ9.eyJhdWQiOiJodHRwczovL3N0cy9hZGZzL29hdXRoMi90b2tlbiIsImV4cCI6MTU1NjY2MzQ5
OCwiaXNzIjoiYmY1MGYyYmQtMTliOS00OTdmLWE1NzUtMDFlODQxNGRmMmY4IiwianRpIjoiM2M2Nzc0YjEtZjI
xNS00NTJkLTg5YzItNjQ5MTZlNjc5ZjZiIiwibmJmIjoxNTU2NjYyODk4LCJzdWIiOiJiZjUwZjJiZC0xOWI5LT
Q5N2YtYTU3NS0wMWU4NDE0ZGYyZjgifQ.dr56QFMA9S9u72XwnaKEEOr0RoPKiTV79HgSs4IDmR0VzgeImqx4KR
up_3gbltiKau_63IYsO1AikPL4cKB6TiTgTUJeJQZok5IBejI5MHw9i6FR7X2btlZy4mEwVr6AJVP0XUP_2lvgR
MlH4TkXkreTwaJo4OqDxToFkcS2kcrZ7TWBhfIocfQvj5FrKS8T3s-pPvdNWiatIr-
```

```
71aXiu41Puke6H2J8NEFgrFc9w4iWZuWt9WUfLfja1RBIbU8K7JUPYkRcuowdv0xj-
lLRJHinjtD0uJex8V02QKCSGMLQ20gLKm8Ez9wlzzwzrf71gE84jaJ3IMrs7oZFB0EJmg&
grant_type=client_credentials&
scope=openid&
resource=dd12c35c-d4d5-465a-9976-8117453f87e6
```

4.1.2.2 Token request parameters

Parameter	Description
client_id	The client ID assigned to you.
client_assertion_type	The value must be set to urn:ietf:params:oauth:client-assertion-type:jwt-bearer.
client_assertion	An assertion (a JSON web token) that you need to create and sign with the certificate registered to your credentials in Active Directory.
grant_type	Must be set to client_credentials.
scope	A space-separated list of scopes. For OpenID Connect, it must include the scope openid.
resource	The provided relying party ID URN of the HES web API (secured resource) to access.

4.1.3 Response

4.1.3.1 Example of token response

```
{
  "access_token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Inp6SXQ2b1BTemEtWm9nRWUwWXRZQnI4QTFVUSJ9.e
yJhdWQiOiJtaWNyb3NvZnQ6aWRlbnRpdHlzZXJ2ZXI6YmY1MGYyYmQtMTliOS00OTdmLWE1NzUtMDFlODQxNGRm
MmY4IiwiaXNzIjoiaHR0cDovL3N0cy9hZGZzL3NlcnZpY2VzL3RydXN0IiwiaWF0IjoxNTU2NjY1MTQzLCJleHA
iOjE1NTY2Njg3NDMsImF1dGhtZXRob2QiOlsiaHR0cDovL3NjaGVtYXMubWljcm9zb2Z0LmNvbS93cy8yMDA4Lz
A2L2lkZW50aXR5L2F1dGhlbnRpY2F0aW9ubWV0aG9kL3Rsc2NsaWVudCIsImh0dHA6Ly9zY2hlbWFzLm1pY3Jvc
29mdC5jb20vd3MvMjAwOC8wNi9pZGVudGl0eS9hdXRoZW50aWNhdGlvbm1ldGhvZC94NTA5Il0sImFwcHR5cGUi
OiJDb25maWRlbnRpYWwiLCJhcHBpZCI6ImJmNTBmMmJkLTE5YjktNDk3Zi1hNTc1LTAxZTg0MTRkZjJmOCIsImF
1dGhfdGltZSI6IjIwMTktMDQtMzBUMjI6NTk6MDMuNjgyWiIsInZlciI6IjEuMCIsInNjcCI6Im9wZW5pZCJ9.M
UOBvrFqCUWwQBO8wc0d3d6jvi8htBEjNfR5GghVNberxR7Qog6beg76YvZBJ0Mh5ZpDC8KspX2HiVRuWekQAZVg
uqW0Rh4_mImY3NLsP9FAIfbVqPYnkEpbr7RTa6z3waYtXBFSQqiPdeiLzNa_LxVL7XB0Yt7pOyywrfSXui045p0
9xgq4JgMI-
wnJbtOASVereFpxj9ac1yy1WZaVHQP1VyZ5VDJQBOleh6x76eFB96VuKDPd4UoZ1xsUKfW4NFWFqgNQY3FqeCkU
QJ4yKhaNAJVWOJSQBovNFwC5n2QCho5F6aof2_NvO8lD1970ZbHcVWo_sCdy8XHGgWKzxw",
  "token_type": "bearer",
  "expires_in": 3600,
  "scope": "openid"
}
```

4.1.3.2 Token response parameters

Parameter	Description
access_token	The requested access token. You can use this token to authenticate to the OMNIA system.
token_type	Indicates the token type value. The only type that Microsoft identity platform supports is bearer.
expires_in	The amount of time that an access token is valid (in seconds).
scope	The value passed for the scope parameter in this request should be the resource identifier for HES assigned to you.

4.1.4 HES request

4.1.4.1 Example of HES request

```
POST /foo
Host: https://hes
Content-Type: text/xml
Authorization: Bearer
eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsIng1dCI6Inp6SXQ2b1BTemEtWm9nRWUwWXRZQnI4QTFVUSJ9.ey
JhdWQiOiJtaWNyb3NvZnQ6aWRlbnRpdHlzZXJ2ZXI6YmY1MGYyYmQtMTliOS00OTdmLWE1NzUtMDFlODQxNGRmM
mY4IiwiaXNzIjoiaHR0cDovL3N0cy9hZGZzL3NlcnZpY2VzL3RydXN0IiwiaWF0IjoxNTU2NjY1MTQzLCJleHAi
OjE1NTY2Njg3NDMsImF1dGhtZXRob2QiOlsiaHR0cDovL3NjaGVtYXMubWljcm9zb2Z0LmNvbS93cy8yMDA4LzA
2L2lkZW50aXR5L2F1dGhlbnRpY2F0aW9ubWV0aG9kL3Rsc2NsaWVudCIsImh0dHA6Ly9zY2hlbWFzLm1pY3Jvc2
9mdC5jb20vd3MvMjAwOC8wNi9pZGVudGl0eS9hdXRoZW50aWNhdGlvbm1ldGhvZC94NTA5Il0sImFwcHR5cGUiO
iJDb25maWRlbnRpYWwiLCJhcHBpZCI6ImJmNTBmMmJkLTE5YjktNDk3Zi1hNTc1LTAxZTg0MTRkZjJmOCIsImF1
dGhfdGltZSI6IjIwMTktMDQtMzBUMjI6NTk6MDMuNjgyWiIsInZlciI6IjEuMCIsInNjcCI6Im9wZW5pZCJ9.MU
OBvrFqCUWwQBO8wc0d3d6jvi8htBEjNfR5GghVNberxR7Qog6beg76YvZBJ0Mh5ZpDC8KspX2HiVRuWekQAZVgu
qW0Rh4_mImY3NLsP9FAIfbVqPYnkEpbr7RTa6z3waYtXBFSQqiPdeiLzNa_LxVL7XB0Yt7pOyywrfSXui045p09
xgq4JgMI-
wnJbtOASVereFpxj9ac1yy1WZaVHQP1VyZ5VDJQBOleh6x76eFB96VuKDPd4UoZ1xsUKfW4NFWFqgNQY3FqeCkU
QJ4yKhaNAJVWOJSQBovNFwC5n2QCho5F6aof2_NvO81D1970ZbHcVWo_sCdy8XHGgWKzxw


<RequestMessage xmlns="http://iec.ch/TC57/2011/schema/message">
  <Header>
    <Verb>get</Verb>
    <Noun>MeterReadings</Noun>
    <Revision>2.0</Revision>
    <Timestamp>2012-10-02T14:16:09Z</Timestamp>
    <AsyncReplyFlag>true</AsyncReplyFlag>
    <ReplyAddress>https://mdms:8090/foobar</ReplyAddress>
    <MessageID>cca4968f-9163-4c8e-8fb6-e43a79a74d06</MessageID>
    <CorrelationID>cca4968f-9163-4c8e-8fb6-e43a79a74d06</CorrelationID>
  </Header>
  <Request>
    <GetMeterReadings xmlns="http://iec.ch/TC57/2011/GetMeterReadings#">
      ...
    </GetMeterReadings>
  </Request>
</RequestMessage>
```