Network Working Group
INTERNET-DRAFT
Intended status: Standards Track
Expires: February 25, 2019

B. Deutsch
Independent Submitter

August 24, 2018

Client Application Layer Encryption
draft-deutch-lamps-client-app-encrypt-00

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Status of This Memo

This document specifies an Experimental protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

Abstract

The protocol for Client Application Layer Encryption offers organizations a method of securely providing users data with very few authentication steps. This protocol makes use of X.509 public key infrastructure and SHOULD NOT be implemented without transport layer security. The protocol described below helps to ensure that response messages may only be read by the intended recipient.

Table Of Contents

Abstract 3
1. Introduction 3
1.1 Terminology 3
1.2 Abbreviations 3
1.3 Roles 3
1.4 Goals 4
1.5 Motivation 4
1.6 Strengths and Weaknesses 4
2. Security Considerations 5

3. IANA Considerations 5
4. Communication Patterns 5
4.1 Initiation 5
4.2 Standard Request 5
4.3 whoami Request 7
4.4 Server Revocation 8
References 8
Normative 8
Informative 8
Appendix A: UML Flow Diagrams 9
A.1 Initiation 9
A.2 Standard Request 10
A.3 whoami Request 11
Appendix B: Example Requests and Responses 12
B.1 Initiation 12
B.2 Standard Request 15
B.3 whoami Request 20
Author's Address 22
Full Copyright Statement 22
Intellectual Property Statement 22

1. Introduction

This protocol offers a way to reduce the number of network communications that must occur for a system to have confidence in the identity of the requester and reduces the risk in the case of impersonation. This was designed with application programming interfaces in mind.

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

1.2 Abbreviations

CN: Common Name [RFC4514]
CSR: certificate signing request [RFC5280]
DN: Distinguished Name [RFC4514]
GUID: Globally Unique IDentifier [RFC4122]
IaaS: infrastructure as a Service
OU: Organizational Unit [RFC4514]
PaaS: Platform as a Service
SAN: subject alternative name [RFC4514]

SaaS: Software as a Service
TLS: transport layer security [RFC5246]

1.3 Roles

resource owner: The party with rights to the data.

resource server: The object housing the data.

authorization server: The server that fulfills certificate signing requests and catalogs them for validation. All calls to this device should be over TLS with mutual certificate exchange [RFC5246].

client: The object requesting the data.

edge device: The object open to anonymous traffic, terminates TLS [RFC5246], brokers authentication, performs authorization, then forwards data.

origination server: The object that performs processing of the request that results in the response.

1.4 Goals

Minimize exposure of client credentials and data. A client can be authorized and returned a token or other sensitive information with confidence that it cannot be intercepted, even by an internal bad actor. To do this the authorization server must either be a signing authority or have permission to submit certificate signing requests to a signing authority [RFC5280]. The client certificate properties may act as a vehicle for personally identifying information to be passed to the origination server. The private key SHOULD NOT be exported from the client device and therefore the CSR may contain device properties.

1.5 Motivation

Organizations have increased the number of individuals with access to subvert trusted systems with the increase in subcontracting information services i.e. SaaS, PaaS, IaaS, etc; as well as contract workers.

When users' information is unencrypted is it vulnerable to exploitation. By reducing the occurrences of client data being

unencrypted we reduce the opportunity for attack.

1.6 Strengths and Weaknesses

This provides a mechanism for user credentials that may be valid for an undefined period of time. Made possible because the credentials themselves, the private key [RFC5280], never exists outside the users' (resource owners') device.

The true proof of identity is in the ability of the client to read the response message. Which makes this mechanism ideal for GET requests but unsuitable for POST, PUT, or DELETE unless accompanied by a secondary authentication mechanism.

If an attacker captured the CSR then they would be in a position to build a response the client would accept, however the attacker would also have to impersonate the edge device in order to impersonate the authorization and origination servers. Conversely, if an attacker impersonates the edge device without the CSR on file then any response would appear malformed.

Because these certificates are not used in TLS negotiation the client is not required to share it at the device layer. This allows the credentials to be owned exclusively by the application within the clients' device, reducing the opportunity for another application running on the same device to steal the private key or impersonate the organization's application to the authorization server by reading their response.

To mitigate risk of attacks some error messages must simulate successful responses reducing feedback to legitimate consumers with malformed requests.

2. Security Considerations

This document defines a protocol for authenticating and authorizing users for access to protected data and the secure delivery of responses.

3. IANA Considerations

No IANA considerations

4. Communication Patterns

The following sections describe the various transactions that make up this protocol.

4.1 Initiation

For this flow the client is also the resource owner, and the authorization server is also both resource server and origination server.

The client must use a method acceptable to the edge device to prove their identity [RFC6749] [RFC7617], preferably initial registration. At the conclusion of this proving the client should have packaged their CSR and sent it to the edge device.

The edge device shall then forward the identity information with the CSR and the cipher used for the TLS to the authorization server.

The authorization server shall store the CSR in association with the user identity and return a response of the GUID of the CSR record encrypted by a certificate generated from the CSR using the cipher negotiated between the client and the edge device. This cipher is used to ensure it is one the client knows, to be sure it is one that the resource server knows; the edge device and resource servers should be configured to maintain the same list (remember in this flow the resource server is also the authorization server).

The edge device shall then return the encrypted response to the client.

The client must decrypt the response with their private key [RFC5280] used to generate the CSR and store the GUID and key for future use.

4.2 Standard Request

For this flow the client is also the resource owner. These credentials are sufficient if this Request is a read only operation or a create that produces data that is only usable after the client has read the response (proving that they are the resource owner), such as token generation where the token is returned in the response payload body or a request to a processing queue which must be followed by an execution call using the queue identifier from the response. These credentials should be supplemented by a secondary mechanism if this request is expected to result in any data changes.

The client shall send their GUID with the request to the edge device.

The edge device should forward the GUID to the authorization server in the form of a validation request. The edge device may forward the request to the origination server without performing this step, which would be bad practice because it increases the opportunity for capture, message replay, and in that case the origination server would need to call the authorization server increasing its client list and therefore attack surface.

The authorization server shall reply to the validation request with a client certificate generated by the CSR associated with the GUID. The certificate should only be valid long enough to fulfill the request.

If the edge device receives a response from the validation call to the authorization server that is not a client certificate then the edge device should return an object large enough to be mistaken for an encrypted response to the untrusted client. If authentication is successful then the edge device should then forward the client request with the certificate and the negotiated cipher to the origination server without the GUID.

If an internal bad actor captured a request with the client's certificate or GUID and used it to send a request then they would be unable to read the response. Additionally, the certificate should have an extremely short validity period in which this request would be valid.

The origination server should validate the certificate by issuer, subject, and expiration. No CRL is required as the certificate validity should only ever be long enough for one request. This enables the origination server to perform fine grained authentication with high confidence without any external calls. The origination server may be or make calls to the resource server(s) providing the certificate and not the cipher, aggregating data as required. The identity of the certificate is taken from the SAN if present; wherein the CN is the resource owner, the DC is the organization of the servers, and any OUs represent allowed scope(s).

The absence of the cipher informs any resource server(s) that their response should not be encrypted by the user's certificate. This request should be over TLS and should use mutual certificate exchange [RFC5246] because the client's certificate in this request

is not for authentication, it is present as a form of query. These requests are from the origination server to the resource server(s) as evidenced by the origination server's need to read the response.

The origination server shall encrypt the response intended for the client using the client's certificate and the cipher provided by the edge device ensuring that only the client is able to decrypt it. The origination server then returns this response to the edge device.

The edge device shall forward the response to the client.

The client shall use their private key to decrypt the response.

If the request is captured between the client and the edge device then a message replay is possible, however the response could only be read by the real client. If a request is captured between the edge device and the origination server then a message replay is possible only until the certificate expires and again, could only be read by the real client. The flow should use TLS throughout to prevent the request from being read between hops.

4.3 whoami Request

For this flow the client is also the resource owner and the authorization server is also both resource server and origination server.

The client makes a request to the edge device using their GUID.

The edge device receives the request and forwards the GUID to the authorization server with the negotiated cipher.

The authorization server generates a certificate for the client that expires immediately, encrypts the certificate using itself and the specified cipher, and then returns this as the response to the edge device. If the GUID is not known then an response consisting of a random salt large enough to be reasonably mistaken for an encrypted payload should be returned to the edge device with a HTTP 200 code [RFC7231], this is intended to prevent a dictionary attack from mapping out valid GUIDs.

The edge device forwards the response to the client.

The resource owner must then decrypt the response to read it.

4.4 Server Revocation

In the event that a set of credentials are compromised then the authorization server may be required to revoke them. The resource owner may be required to perform a new initiation to regain access to their account.

References

Normative

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Level", BCP 14, RFC 2119, March 1997.

[RFC5246] Dierks, T., "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5280] Cooper, D., "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

Informative

[RFC4122] Leach, P., "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, July 2005.

[RFC4514] Zeilenga, K., "Lightweight Directory Access Protocol (LDAP): String Representation of Distinguished Names", RFC 4514, June 2006.

[RFC5751] Ramsdell, B., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification", RFC 5751, January 2010.

[RFC6749] Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012.

[RFC7231] Fielding, R., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, June 2014.

[RFC7617] Reschke, J., "The 'Basic' HTTP Authentication Scheme", RFC 7617, September 2015.

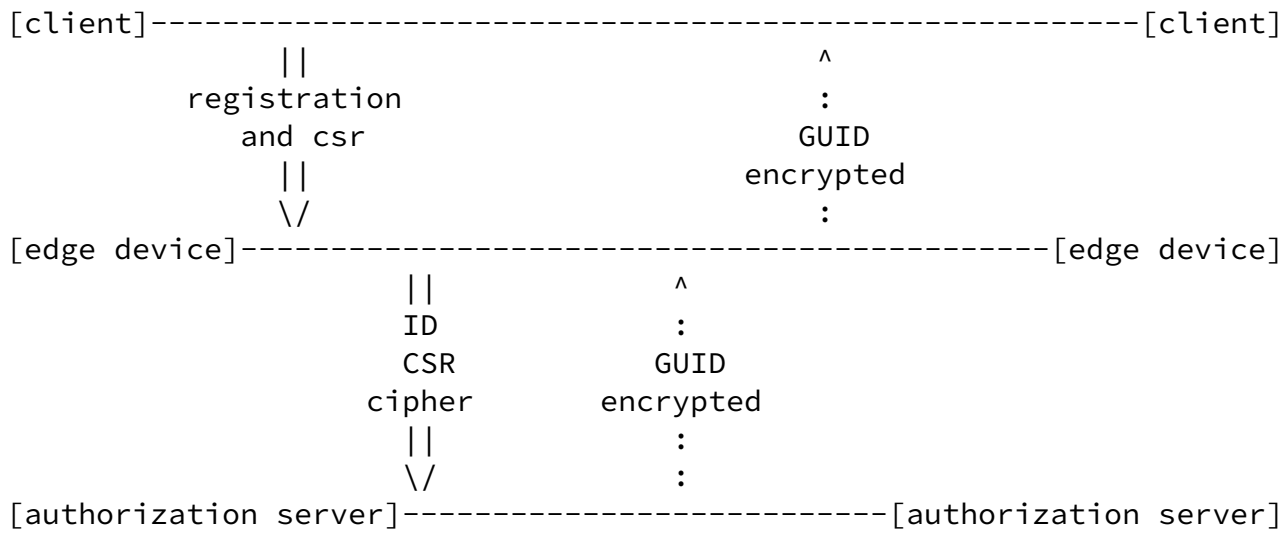
[WSD] WebSequenceDiagrams software is provided by Hanov Solutions Inc., of Waterloo, Ontario, Canada. <https://www.websequencediagrams.com>

Appendix A: UML Flow Diagrams

Each section of this appendix corresponds to the same numbered sub section of this document under section 4. The text between the section heading and the flow graphic represents the flow in sudo-code [WSD]. The diagrams have been simplified from the sudo-code in order to fit this document format.

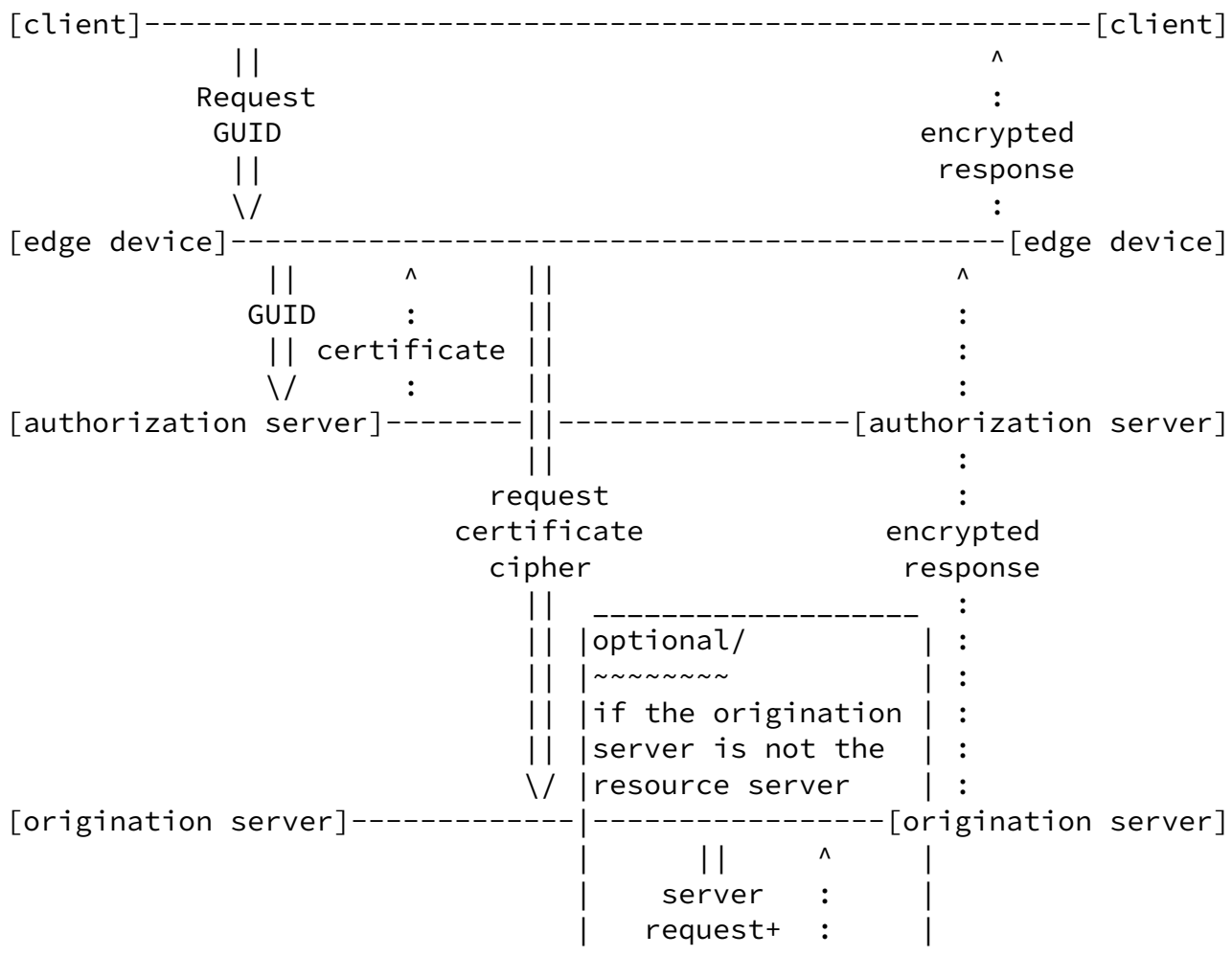
A.1 Initiation

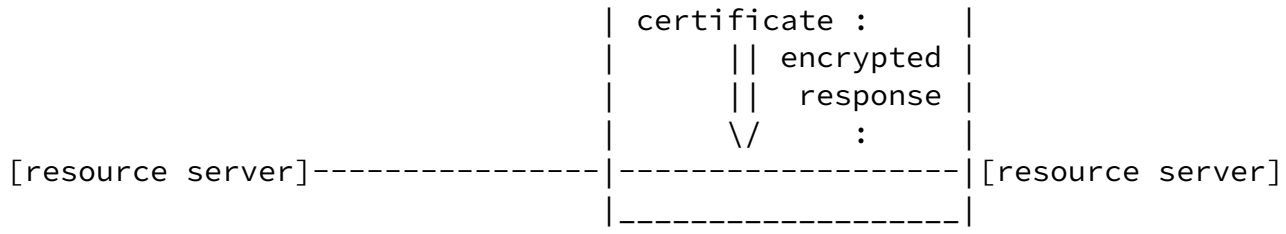
```
    title Initiation

    note over client:
        generate key
        generate CSR
    end note
    client->edge device: Registration+CSR
    edge device->+authorization server: ID+CSR+cipher
    note over authorization server:
        store CSR
        generate GUID
        encrypt GUID
    end note
    authorization server-->-edge device: encrypted GUID
    edge device-->+client: encrypted GUID
    note over client:
        decrypt response
        store GUID
    end note
```



A.2 Standard Request

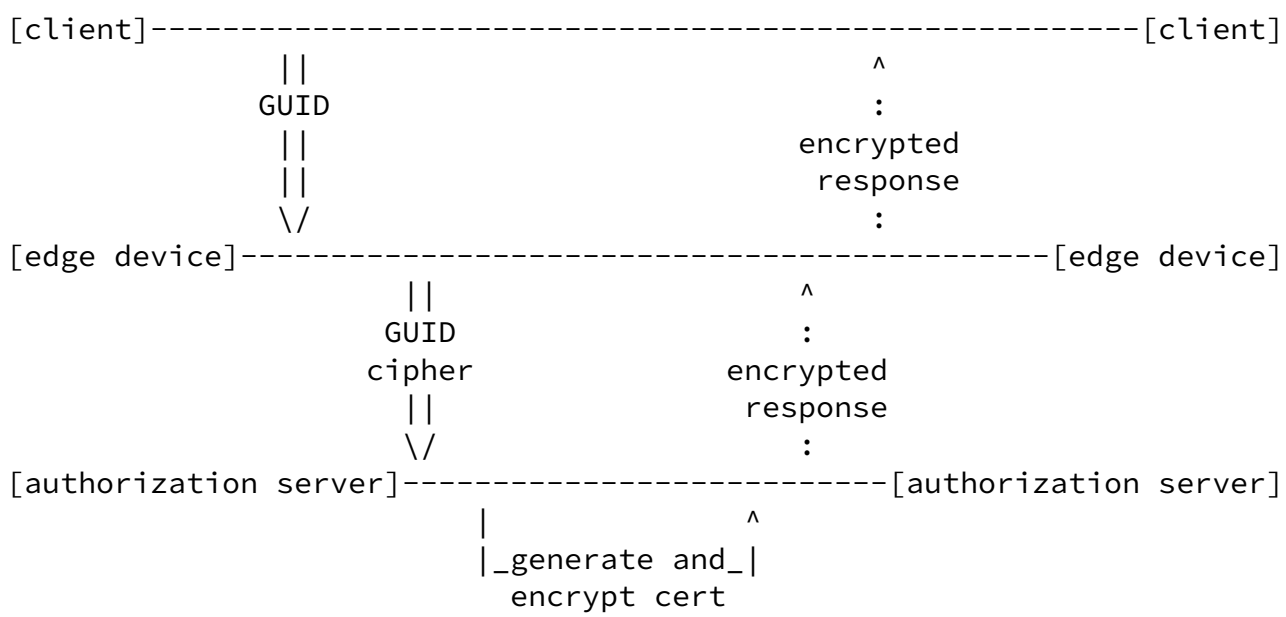
```
    title Standard Request

    client->edge device: request+GUID
    edge device->authorization server: GUID
    note over authorization server: generate certificate from GUID CSR
    authorization server-->edge device: certificate
    edge device->origination server: request+certificate+cipher
    note over origination server:
        certificate validation
        authorization
    end note
    opt if origination server is not resource server
        origination server->resource server: server request+certificate
        resource server-->origination server: server response
    end
    note over origination server: encrypt response
    origination server-->edge device: encrypted response
    edge device-->client: encrypted response
    note over client: decrypt response
```





A.3 whoami Request

```
    title whoami Request

    client->edge device: GUID
    edge device->authorization server: GUID+cipher
    note over authorization server:
        generate certificate
        encrypt response
    end note
    authorization server-->edge device: encrypted response
    edge device-->client: encrypted response
    note over client: decrypt response
```



Appendix B: Example Requests and Responses

Each section of this appendix corresponds to the same numbered sub section of this document under section 4. These examples contain elements which fulfill the requirements described above and may be met by other means.

B.1 Initiation

The below private key is used to generate the below examples and can be used to execute the client decryption commands:

```
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,7F58E7878FA4D4A8

98MDLxjgMW5W71ZADD1CG2VeAMG/vxmIqpF+2japv831iSh4WC5LJfPXfKXp+nQ5
L74+xDt1fRSKuPfnBqPnok1lZrMqK+WtW83FSxA0wm5Rvfsa9ECSoMJP24z5roAd
+ipyn3v47Vmlu6gjk1wmgj2hT2LnkrwvXh6CGKc2AjA3xQieGKyzB6/m2hMc6A5z
nVwhwJi4Fc4J/Zs9+J/4KUFbSdobrs5Ej4iexWWTGfzVDjOmRaObol6gxldiGDH3
1khUSpJutOjnisiWUVkjUos9AvFi8QISeodiQr5AMCrYGVY0N5BN35hv/mqJHT3g
AH25psCwaT1P44qYu6CQSRkRxOE2CmJIhPvsPjC9uOx8zois3ICwCUZLUpkwsL7o
DfnucGNPpS5aIJenno5Cy8aY5E0BXN/m5OxLfciWAp4Sv2Fg55TKDLaysIcHNy9G
J56SD2QJEoF7s9LbUykGutlBOI1ozWxyhMK2ku/DjB0lQTncUaibWJ5Y3Bw1uVVe
8GL3HAoR8G+aos8ESy/0vcaEHmTM4iqXKZcRELvlGJ+HqCSalLVgf8XaLMNPq+8l
qTEHPGPVpo5BQRLCavg21hd38nBmFHQFyB5X6jQcAhBuhf8Ns12Na72V4OHyXtKD
hB/qfrdQukkAhCRRGFbsc3iOwM/OkUwv/z/w1NikP6Z9jhTQIC/RF/86CGAEyfdQ
1FC+wsDgkbmKaQIWoyqdrHiXiLI2htMSJ4aeJufjFvH86PhZiEi0gUKpkFqi58ix
0kfoiUO3oAEPKAoZvGDlRN+/x89hjccqmOFoKDxckUaKphTzuJwepQDNaRkPSNKd
d6yjD4nB9Bjnbp1bwR/iy7OCA33lRangFuUFq6gsZsj54Ioi8MOZ8aox0GdrM8so
eWexF7od+L6/zBh43WHE0vMDjOUX8QgkCXWF1mhP0Yd13uLsddaUeYtjDJp44t2y
pb8FdnfA5IS7xMyNz2XIBZJOtqSGrWvPR9o/xloZiuNBOzmns6wmz3ZGznZddVex
s9nM2VoPdrPe8n4bxuTRXPyGvATDdY8czqZh8/STGX5PPmCvRA1ilWrN1sP844mq
QSv1swG+bnDIgAZS9D7DR5pq5Ed18Zby5g6l0uUwEDQeIonMsRwHErQtB2X3rMX+
lHg35WKHTRjPk6kcGWWcRuBkHmKSug4qDqjbQZLNaBA9v2XxB5CuoJ8yFMGRz8oO
phflxeJWA8w185UPQ9Sm8m/S6nP5NjdO4XUzzhJ7Ue/+Um2XrghRfY1+mGDo+B5a
PPvKf2VetChVXIpFew1fZWfQuZGluJTHdb1J7lG2Q9rKrLY7ty0P+gMQs8by8nwm
XYgJiqXnzr15u005JQpXhkfJ1B4x+0K5q1vVJNenlvLa40r+/wU8tNFEV9cgBtPm
B4+Zikt+FD2A1uU+9wCOBanXE/xCN95oTCH06FMiv8j/qzh9+c7DnNxPQ8rvCQf1
dH8A1kMxgOJ9zIfuZMmAUMQmI3t5qh4oGT8RycWa/e1JeMxiMqpOSY7cwH5UyzpM
/8ZrWLpPo7CYnTvK4LaMBzhvu6mxp348dNR8qmxIkmH7rcqXyPu+BVwTpt/2pXVe
-----END RSA PRIVATE KEY-----
```


B.1.1 Client Registration Request

The client generates a private key:

```
openssl genrsa -des3 -out privkey.key 2048
```

Then generate a certificate from the key to designate the expected properties:

```
openssl req -key privkey.key -out client.crt -new -x509
```

Then generates a CSR from the key and certificate:

```
openssl x509 -x509toreq -in client.crt -out client.req -sha256
-signkey privkey.key
```

The request from the client to the Edge Device:

```
POST /registration HTTP/1.1
Host: server.example.com
Content-Type: application/pkcs10

-----BEGIN CERTIFICATE REQUEST-----
MIICrzCCAZcCAQAwajELMAkGA1UEBhMCVVMxCzAJBgNVBAgMAkNBMRUwEwYDVQQH
DAxEZWZhdWx0IENpdHkxHDAaBgNVBAoME0RlZmF1bHQgQ29tcGFueSBMdGQxGTAX
BgNVBAMMEEJlbmphbWluIERldXRzY2gwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAw
ggEKAoIBAQCpJMQQY1gzANViIIreVQvIlp2mw1ASUixRJp4SGPHpsaNJfvHcZBWl
zBfVfh960OsC1NasUs69WQIPeuJAYELdOXYox2J+5DSN/g3X8p3CXMrVd7xpArpx
q6uxevEtMP1kx4X8VC7nJsEPJO1lFhwTixWuUQv5xWL5qGuATafmtRvbBWNBMRa8
55HCKIcQkx4i0/DMREm0P/7fYRfuwYUWf3KJfkuCnwhbmxvFI0PDQfw/q+UhpobV
arxZS++S6jlMdaKh7tHLOLpfHdrLr8uaNlOB3weF6C2EGDxlzB0v3xEmdxVL7Ch6
GBZ7y3amfydZ5FOK1SD3lgWWYMm/6E5tAgMBAAGgADANBgkqhkiG9w0BAQsFAAOC
AQEAUnKJBIenLImXFBl7J3GwL948KPbKnuc7HRex0TmSo4G7fN7RxIo+6uZEgFG0
met55u+5uepVyGYnph2tgwO7hYUnUA5Zl4fzJeNmXljBAfBUQ4DYhi6R5yCpzU1C
wJOSyWWujPPUvfsRnT5kbk7LBvHKqntZ8+s3mbUtVVb80VsaWvOzDZerS6K+OXnY
YpV4oqZOmhraYDDtFuGVWBYJNspZwjNHTXJjhgR0u+xhnX8PugIoULIan/SmFkt/
6pvIjgOBX1NbBQo4B8S1F+l6R9CShEX6UCALkd+9BhHXDDiTZZara1YshpOEFr9W
qMHUCVVDTcYZomsqQqU/wKF8Hg==
-----END CERTIFICATE REQUEST-----
```


B.1.2 Edge Registration Request

The request forwarded to the authorization server with the cipher:

```
POST /registration HTTP/1.1
Host: server.example.com
Content-Type: application/pkcs10
Cipher: ECDHE-RSA-AES256-SHA

-----BEGIN CERTIFICATE REQUEST-----
MIICrzCCAZcCAQAwajELMAkGA1UEBhMCVVMxCzAJBgNVBAgMAkNBMRUwEwYDVQQH
DAxEZWZhdWx0IENpdHkxHDAaBgNVBAoME0RlZmF1bHQgQ29tcGFueSBMdGQxGTAX
```

```
BgNVBAMMEEJlbmphbWluIERldXRzY2gwggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAw
ggEKAoIBAQCpJMQQY1gzANViIIreVQvIlp2mw1ASUixRJp4SGPHpsaNJfvHcZBWl
zBfVfh960OsC1NasUs69WQIPeuJAYELdOXYox2J+5DSN/g3X8p3CXMrVd7xpArpx
q6uxevEtMP1kx4X8VC7nJsEPJO1lFhwTixWuUQv5xWL5qGuATafmtRvbBWNBMRa8
55HCKIcQkx4i0/DMREm0P/7fYRfuwYUWf3KJfkuCnwhbmxvFI0PDQfw/q+UhpobV
arxZS++S6jlMdaKh7tHLOLpfHdrLr8uaNlOB3weF6C2EGDxlzB0v3xEmdxVL7Ch6
GBZ7y3amfydZ5FOK1SD3lgWWYMm/6E5tAgMBAAGgADANBgkqhkiG9w0BAQsFAAOC
AQEAUnKJBIenLImXFBl7J3GwL948KPbKnuc7HRex0TmSo4G7fN7RxIo+6uZEgFG0
met55u+5uepVyGYnph2tgwO7hYUnUA5Zl4fzJeNmXljBAfBUQ4DYhi6R5yCpzU1C
wJOSyWWujPPUvfsRnT5kbk7LBvHKqntZ8+s3mbUtVVb80VsaWvOzDZerS6K+OXnY
YpV4oqZOmhraYDDtFuGVWBYJNspZwjNHTXJjhgR0u+xhnX8PugIoULIan/SmFkt/
6pvIjgOBX1NbBQo4B8S1F+l6R9CShEX6UCALkd+9BhHXDDiTZZara1YshpOEFr9W
qMHUCVVDTcYZomsqQqU/wKF8Hg==
-----END CERTIFICATE REQUEST-----
```


B.1.3 Registration Response

After generating the GUID that identifies the record it shall be encoded using the client certificate:

```
openssl smime -encrypt -binary -aes-256-cbc -in response.txt
client.crt
```

Resulting in the encrypted response [RFC5751]:

```
HTTP/1.1 200 OK
Content-Type: text/plain;charset=UTF-8
MIME-Version: 1.0
Content-Disposition: attachment; filename="smime.p7m"
Content-Type: application/x-pkcs7-mime;
smime-type=enveloped-data; name="smime.p7m"
Content-Transfer-Encoding: base64

MIICCwYJKoZIhvcNAQcDoIIB/DCCAfgCAQAxggGTMIIBjwIBADB3MGoxCzAJBgNV
BAYTAlVTMQswCQYDVQQIDAJDQTEVMBMGA1UEBwwMRGVmYXVsdCBDaXR5MRwwGgYD
VQQKDBNEZWZhdWx0IENvbXBhbnkgTHRkMRkwFwYDVQQDDBBCZW5qYW1pbiBEZXV0
c2NoAgkAondW3eFlchkwDQYJKoZIhvcNAQEBBQAEggEACddDSDsbQ5D+eMwSqpNa
XHQOI1nWEYBDTx294ub67XV8ZxKGnMi/zMlSvdsNTlhXhz5/TjN8vwGF7v30znXM
4fvUXQpCOps8APG5y3tWe8I7XPTKsTtaJymCDAoBokLIIFfjgMo6Yh3qDZ53PSdG
wN2WxDlhAFyob6lX2WTPzh5RlCSmbWwEt3AnZqshHxLs8uk7ci3BU9Coizw3lVBh
vcH5hH6A8ad1bE4y+s3SRrPqTag4/CXz/LXC9i5WrMbXqVz6yKnH1CgkX4k0NMbe
DqjHnsUV7M1TuHfb+NFI329bOQKofqIIVseq4S7rIzpbrEsDehPZt5kwMxTOttUX
YzBcBgkqhkiG9w0BBwEwHQYJYIZIAWUDBAEqBBCZc4CRchSYISroxg0r6twPgDCK
WSrODqmsS8zckitZgLcftiZ2hsGbmCUiq5pUwZdEBmMzGJIIl4w+mLmTYuhKOHU=
```

The client decrypts the response:

```
openssl smime -decrypt -binary -aes-256-cbc -in response.enc -out
response.txt -inkey privkey.key
Enter pass phrase for privkey.key: password

bec6dc7e-6562-4b1c-b308-6c352e6f8404
```

B.2 Standard Request

A request to some other services with this added protection.

B.2.1 Standard Client Request

The request to some service:

```
GET /resource HTTP/1.1
Host: server.example.com
CALE-GUID: bec6dc7e-6562-4b1c-b308-6c352e6f8404
```

B.2.2 Edge Validation Request

The authentication request to the authorization server:

```
GET /validate HTTP/1.1
Host: authority.example.com
CALE-GUID: bec6dc7e-6562-4b1c-b308-6c352e6f8404
```

B.2.3a Authorization Validation Response

Create the signed certificate with minimally applicable validity:

```
openssl ca -config openssl.cnf -startdate 180731190800Z -enddate
180731190810Z -keyfile ca.key -cert ca.crt -in client.req -out
./client.crt -notext
```

A successful response from the authorization server:

```
HTTP/1.1 200 OK
Content-Type: application/x509

-----BEGIN CERTIFICATE-----
MIIEfDCCA2SgAwIBAgIRAOaxLLnaTZDrituxMDU+EwowDQYJKoZIhvcNAQELBQAw
czELMAkGA1UEBhMCVVMxCzAJBgNVBAgMAkNBMRUwEwYDVQQHDAxEZWZhdWx0IENp
dHkxFDASBgNVBAoMC2V4YW1wbGUuY29tMQswCQYDVQQLDAJJVDEdMBsGA1UEAwwU
YXV0aG9yaXphdGlvbiBzZXJ2ZXIwHhcNMTgwNzMxMTkwODAwWhcNMTgwNzMxMTkw
ODEwWjBqMQswCQYDVQQGEwJVUzELMAkGA1UECAwCQ0ExFTATBgNVBAcMDERlZmF1
bHQgQ2l0eTEcMBoGA1UECgwTRGVmYXVsdCBDb21wYW55IEx0ZDEZMBcGA1UEAwwQ
```

```
QmVuamFtaW4gRGV1dHNjaDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
AKkkxBBjWDMA1WIgit5VC8iWnabDUBJSLFEmnhIY8emxo0l+8dxkFaXMF9V+H3rQ
6wLU1qxSzr1ZAg964kBgQt05dijHYn7kNI3+DdfyncJcytV3vGkCunGrq7F68S0w
/WTHhfxULucmwQ8k7WUWHBOLFa5RC/nFYvmoa4BNp+a1G9sFY0ExFrznkcIohxCT
HiLT8MxESbQ//t9hF+7BhRZ/col+S4KfCFubG8UjQ8NB/D+r5SGmhtVqvFlL75Lq
OUx1oqHu0cs4ul8d2suvy5o2U4HfB4XoLYQYPGXMHS/fESZ3FUvsKHoYFnvLdqZ/
J1nkU4rVIPeWBZZgyb/oTm0CAwEAAaOCARIwggEOMAkGA1UdEwQCMAAwDgYDVR0P
AQH/BAQDAgWgMB0GA1UdJQQWMBQGCCsGAQUFBwMBBggrBgEFBQcDAjAdBgNVHQ4E
FgQUSxqn9ioM+4Im9NWszrg3xvB3Xt4wHwYDVR0jBBgwFoAU8/02wa7539I+EYiE
mgMYyLFLHfwwZAYIKwYBBQUHAQEEWDBWMCgGCCsGAQUFBzAChhxodHRwOi8vY2Eu
c2FtcGxlLmxhbi9jYS5odG1sMCoGCCsGAQUFBzABhh5odHRwOi8vb2NzcC5jYS5z
YW1wbGUubGFuOjkwODAwLAYDVR0fBCUwIzAhoB+gHYYbaHR0cDovL2NhLnNhbXBs
ZS5sYW4vY2EuY3JsMA0GCSqGSIb3DQEBCwUAA4IBAQDM1uhIypvCU+w0ZyW4fTXg
Zmpp/S3HoFvthVYVfnI5fhUumntFtRQHGyi468qH1Q79UGXW3wnx4Mz//2xQamRu
ACv16+pDXlMxrNJPk5udSHyweqESiaQS1wYqkMsVKx7Sk2AMH8c8cWoUZkBB62ZG
rQMAT0XHP9l/b7qnqNmgS/YkFNfl7uK1FTWLSzGUfVSoFD6YAtLpP0jfgZy+hy69
eG5dRrkagxT22tK9+o+DFSGMhsIQI++UDMypCRjyFQgmWXMj4DW1olZz7u90eQCT
WfSkZ+Elpp19Xmboki4KVriVJm2zMZN/1+sxcWpLe2BHAxb3V+erkwNMt+wog/kS
-----END CERTIFICATE-----
```

B.2.3b authentication Validation Error

An unsuccessful response from the authorization server:

```
HTTP/1.1 403 Forbidden
```


B.2.3c Edge Device Erroneous Response

A successful appearing response designed to prevent dictionary attack from mapping real user GUIDs (mocking B.2.7).

```
HTTP/1.1 200 OK
Content-Type: text/plain;charset=UTF-8
MIME-Version: 1.0
Content-Disposition: attachment; filename="smime.p7m"
Content-Type: application/x-pkcs7-mime;
smime-type=enveloped-data; name="smime.p7m"
Content-Transfer-Encoding: base64

QXQgdmVybyBlb3MgZXQgYWNjdXNhbXVzIGV0IGl1c3RvIG9kaW8gZGlnbmlzc2lt
b3MgZHVjaW11cyBxdWkgYmxhbmRpdGlpcyBwcmFlc2VudGl1bSB2b2x1cHRhdHVt
IGRlbGVuaXRpIGF0cXVlIGNvcnJ1cHRpIHF1b3MgZG9sb3JlcyBldCBxdWFzIG1v
bGVzdGlhcyBleGNlcHR1cmkgc2ludCBvY2NhZWNhdGkgY3VwaWRpdGF0ZSBub24g
cHJvdmlkZW50LCBzaW1pbGlxdWUgc3VudCBpbiBjdWxwYSBxdWkgb2ZmaWNpYSBk
```

```
ZXNlcnVudCBtb2xsaXRpYSBhbmltaSwgaWQgZXN0IGxhYm9ydW0gZXQgZG9sb3J1
bSBmdWdhLiBFdCBoYXJ1bSBxdWlkZW0gcmVydW0gZmFjaWxpcyBlc3QgZXQgZXhw
ZWRpdGEgZGlzdGluY3Rpby4gTmFtIGxpYmVybyB0ZW1wb3JlLCBjdW0gc29sdXRh
IG5vYmlzIGVzdCBlbGlnZW5kaSBvcHRpbyBjdW1xdWUgbmloaWwgaW1wZWRpdCBx
dW8gbWludXMgaWQgcXVvZCBtYXhpbWUgcGxhY2VhdCBmYWNlcmUgcG9zc2ltdXMs
IG9tbmlzIHZvbHVwdGFzIGFzc3VtZW5kYSBlc3QsIG9tbmlzIGRvbG9yIHJlcGVs
bGVuZHVzLiBUZW1wb3JpYnVzIGF1dGVtIHF1aWJ1c2RhbSBldCBhdXQgb2ZmaWNp
aXMgZGViaXRpcyBhdXQgcmVydW0gbmVjZXNzaXRhdGlidXMgc2FlcGUgZXZlbmll
dCB1dCBldCB2b2x1cHRhdGVzIHJlcHVkaWFuZGFlIHNpbnQgZXQgbW9sZXN0aWFl
IG5vbiByZWN1c2FuZGFlLiBJdGFxdWUgZWFydW0gcmVydW0gaGljIHRlbmV0dXIg
YSBzYXBpZW50ZSBkZWxlY3R1cywgdXQgYXV0IHJlaWNpZW5kaXMgdm9sdXB0YXRp
YnVzIG1haW9yZXMgYWxpYXMgY29uc2VxdWF0dXIgYXV0IHBlcmZlcmVuZGlzIGRv
bG9yaWJ1cyBhc3BlcmlvcmVzIHJlcGVsbGF0Lg==
```


B.2.4 Edge Forwarded Request

The request to some service:

```
GET /resource HTTP/1.1
Host: server.example.com
CALE-PEM: "MIIEfDCCA2SgAwIBAgIRAOaxLLnaTZDrituxMDU+EwowDQYJKoZIhvc
NAQELBQAwczELMAkGA1UEBhMCVVMxCzAJBgNVBAgMAkNBMRUwEwYDVQQHDAxEZWZhd
Wx0IENpdHkxFDASBgNVBAoMC2V4YW1wbGUuY29tMQswCQYDVQQLDAJJVDEdMBsGA1U
EAwwUYXV0aG9yaXphdGlvbiBzZXJ2ZXIwHhcNMTgwNzMxMTkwODAwWhcNMTgwNzMxM
TkwODEwWjBqMQswCQYDVQQGEwJVUzELMAkGA1UECAwCQ0ExFTATBgNVBAcMDERlZmF
1bHQgQ2l0eTEcMBoGA1UECgwTRGVmYXVsdCBDb21wYW55IEx0ZDEZMBcGA1UEAwwQQ
mVuamFtaW4gRGV1dHNjaDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAKk
kxBBjWDMA1WIgit5VC8iWnabDUBJSLFEmnhIY8emxo0l+8dxkFaXMF9V+H3rQ6wLU1
qxSzr1ZAg964kBgQt05dijHYn7kNI3+DdfyncJcytV3vGkCunGrq7F68S0w/WTHhfx
ULucmwQ8k7WUWHBOLFa5RC/nFYvmoa4BNp+a1G9sFY0ExFrznkcIohxCTHiLT8MxES
bQ//t9hF+7BhRZ/col+S4KfCFubG8UjQ8NB/D+r5SGmhtVqvFlL75LqOUx1oqHu0cs
4ul8d2suvy5o2U4HfB4XoLYQYPGXMHS/fESZ3FUvsKHoYFnvLdqZ/J1nkU4rVIPeWB
ZZgyb/oTm0CAwEAAaOCARIwggEOMAkGA1UdEwQCMAAwDgYDVR0PAQH/BAQDAgWgMB0
GA1UdJQQWMBQGCCsGAQUFBwMBBggrBgEFBQcDAjAdBgNVHQ4EFgQUSxqn9ioM+4Im9
NWszrg3xvB3Xt4wHwYDVR0jBBgwFoAU8/02wa7539I+EYiEmgMYyLFLHfwwZAYIKwY
BBQUHAQEEWDBWMCgGCCsGAQUFBzAChhxodHRwOi8vY2Euc2FtcGxlLmxhbi9jYS5od
G1sMCoGCCsGAQUFBzABhh5odHRwOi8vb2NzcC5jYS5zYW1wbGUubGFuOjkwODAwLAY
DVR0fBCUwIzAhoB+gHYYbaHR0cDovL2NhLnNhbXBsZS5sYW4vY2EuY3JsMA0GCSqGS
Ib3DQEBCwUAA4IBAQDM1uhIypvCU+w0ZyW4fTXgZmpp/S3HoFvthVYVfnI5fhUumnt
FtRQHGyi468qH1Q79UGXW3wnx4Mz//2xQamRuACv16+pDXlMxrNJPk5udSHyweqESi
aQS1wYqkMsVKx7Sk2AMH8c8cWoUZkBB62ZGrQMAT0XHP9l/b7qnqNmgS/YkFNfl7uK
1FTWLSzGUfVSoFD6YAtLpP0jfgZy+hy69eG5dRrkagxT22tK9+o+DFSGMhsIQI++UD
MypCRjyFQgmWXMj4DW1olZz7u90eQCTWfSkZ+Elpp19Xmboki4KVriVJm2zMZN/1+s
xcWpLe2BHAxb3V+erkwNMt+wog/kS"
Cipher: ECDHE-RSA-AES256-SHA
```

B.2.5 Aggregation Request

A request from the origin server to another resource server:

```
GET /aggregate HTTP/1.1
Host: origin.example.com
CALE-PEM: "MIIEfDCCA2SgAwIBAgIRAOaxLLnaTZDrituxMDU+EwowDQYJKoZIhvc
NAQELBQAwczELMAkGA1UEBhMCVVMxCzAJBgNVBAgMAkNBMRUwEwYDVQQHDAxEZWZhd
Wx0IENpdHkxFDASBgNVBAoMC2V4YW1wbGUuY29tMQswCQYDVQQLDAJJVDEdMBsGA1U
EAwwUYXV0aG9yaXphdGlvbiBzZXJ2ZXIwHhcNMTgwNzMxMTkwODAwWhcNMTgwNzMxM
TkwODEwWjBqMQswCQYDVQQGEwJVUzELMAkGA1UECAwCQ0ExFTATBgNVBAcMDERlZmF
1bHQgQ2l0eTEcMBoGA1UECgwTRGVmYXVsdCBDb21wYW55IEx0ZDEZMBcGA1UEAwwQQ
mVuamFtaW4gRGV1dHNjaDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAKk
kxBBjWDMA1WIgit5VC8iWnabDUBJSLFEmnhIY8emxo0l+8dxkFaXMF9V+H3rQ6wLU1
qxSzr1ZAg964kBgQt05dijHYn7kNI3+DdfyncJcytV3vGkCunGrq7F68S0w/WTHhfx
ULucmwQ8k7WUWHBOLFa5RC/nFYvmoa4BNp+a1G9sFY0ExFrznkcIohxCTHiLT8MxES
bQ//t9hF+7BhRZ/col+S4KfCFubG8UjQ8NB/D+r5SGmhtVqvFlL75LqOUx1oqHu0cs
4ul8d2suvy5o2U4HfB4XoLYQYPGXMHS/fESZ3FUvsKHoYFnvLdqZ/J1nkU4rVIPeWB
ZZgyb/oTm0CAwEAAaOCARIwggEOMAkGA1UdEwQCMAAwDgYDVR0PAQH/BAQDAgWgMB0
GA1UdJQQWMBQGCCsGAQUFBwMBBggrBgEFBQcDAjAdBgNVHQ4EFgQUSxqn9ioM+4Im9
NWszrg3xvB3Xt4wHwYDVR0jBBgwFoAU8/02wa7539I+EYiEmgMYyLFLHfwwZAYIKwY
BBQUHAQEEWDBWMCgGCCsGAQUFBzAChhxodHRwOi8vY2Euc2FtcGxlLmxhbi9jYS5od
G1sMCoGCCsGAQUFBzABhh5odHRwOi8vb2NzcC5jYS5zYW1wbGUubGFuOjkwODAwLAY
DVR0fBCUwIzAhoB+gHYYbaHR0cDovL2NhLnNhbXBsZS5sYW4vY2EuY3JsMA0GCSqGS
Ib3DQEBCwUAA4IBAQDM1uhIypvCU+w0ZyW4fTXgZmpp/S3HoFvthVYVfnI5fhUumnt
FtRQHGyi468qH1Q79UGXW3wnx4Mz//2xQamRuACv16+pDXlMxrNJPk5udSHyweqESi
aQS1wYqkMsVKx7Sk2AMH8c8cWoUZkBB62ZGrQMAT0XHP9l/b7qnqNmgS/YkFNfl7uK
1FTWLSzGUfVSoFD6YAtLpP0jfgZy+hy69eG5dRrkagxT22tK9+o+DFSGMhsIQI++UD
MypCRjyFQgmWXMj4DW1olZz7u90eQCTWfSkZ+Elpp19Xmboki4KVriVJm2zMZN/1+s
xcWpLe2BHAxb3V+erkwNMt+wog/kS"
```

B.2.6 Aggregation Response

A response from a resource server to the origin server:

```
HTTP/1.1 200 OK

{"foo": "bar"}
```

B.2.7 Origination Response

The encrypted response from the origination server that will be passed back to the client by the edge device:

```
HTTP/1.1 200 OK
Content-Type: text/plain;charset=UTF-8
MIME-Version: 1.0
Content-Disposition: attachment; filename="smime.p7m"
Content-Type: application/x-pkcs7-mime;
smime-type=enveloped-data; name="smime.p7m"
Content-Transfer-Encoding: base64

MIIB6wYJKoZIhvcNAQcDoIIB3DCCAdgCAQAxggGTMIIBjwIBADB3MGoxCzAJBgNV
BAYTAlVTMQswCQYDVQQIDAJDQTEVMBMGA1UEBwwMRGVmYXVsdCBDaXR5MRwwGgYD
VQQKDBNEZWZhdWx0IENvbXBhbnkgTHRkMRkwFwYDVQQDDBBCZW5qYW1pbiBEZXV0
c2NoAgkAondW3eFlchkwDQYJKoZIhvcNAQEBBQAEggEAJYwQ+oFA8nm4sp/crwHi
BY1+oVwqnygrXu4aZibJBA5qXQPYYVKGmjgZ1HnvtgWPdV4EW0b3FHbhI71fvalQ
HI3g7Jl9bcyNP0kSt4XmuAZzKrVRktBcEbhP9ePqAoH5S0u4vhwtKMZ/rt0BUPwY
ZQxVAQo7HQDL00+LHu2nGAbVinszn/5bQrJ7CTHO72ecs7m9DBJmaOT+ZT8toEpI
9zOvE4Z6AsqbbrthvIAApWfNBLYxm6fgy+5XeYPdwNnaAOMC0XXEWolp1/Suchzf
f84z7ayH8Xx6cP5mZQe/LH5KT4CvfxwsfhzVkMJkUOKyU7uxA+6B6lqm3t1mgIwy
EjA8BgkqhkiG9w0BBwEwHQYJYIZIAWUDBAEqBBA7pyAHv3GrWkoZc5fiYkBLgBBL
JQYQttSM00rzK3y5X/sA
```

B.3 whoami Request

B.3.1 Client whoami Request

The request to some service:

```
GET /whoami HTTP/1.1
Host: server.example.com
CALE-GUID: bec6dc7e-6562-4b1c-b308-6c352e6f8404
```

B.3.2 Edge whoami Request

The request forwarded to the authorization server with the cipher:

```
GET /whoami HTTP/1.1
Host: server.example.com
CALE-GUID: bec6dc7e-6562-4b1c-b308-6c352e6f8404
Cipher: ECDHE-RSA-AES256-SHA
```


B.3.3 Authorization whoami Response

Generate the certificate that expires immediately:

```
openssl x509 -req -days 0 -in client.req -CA ca.crt -CAkey
ca.key -CAserial file.srl -out client.pem
```

The certificate is encrypted with itself using the cipher:

```
openssl smime -encrypt -binary -aes-256-cbc -in client.pem
client.pem
```

```
HTTP/1.1 200 OK
MIME-Version: 1.0
Content-Disposition: attachment; filename="smime.p7m"
Content-Type: application/x-pkcs7-mime;
smime-type=enveloped-data; name="smime.p7m"
Content-Transfer-Encoding: base64

MIIGogYJKoZIhvcNAQcDoIIGkzCCBo8CAQAxggGWMIIBkgIBADB6MHMxCzAJBgNV
BAYTAlVTMQswCQYDVQQIDAJDQTEVMBMGA1UEBwwMRGVmYXVsdCBDaXR5MRQwEgYD
VQQKDAtleGFtcGxlLmNvbTELMAkGA1UECwwCSVQxHTAbBgNVBAMMFGF1dGhvcml6
YXRpb24gc2VydmVyAgMAhkkwDQYJKoZIhvcNAQEBBQAEggEAGbGnIDFMlf28nPpc
lN7RPb8Ok03T+wESUVDi3Pl9WEiE5BlV00jFPPZYJtatelJt7HOjK0b6Irz5ZHJ6
nzZ3xUN1nOOGPl8E/zffxfmVwBX6mh9jLZSZcPoorM58vUT1a0ci4euMH8pLQ+lZ
t1K+iV9bLm7Bg/xqumyhjrMq+lb5+0a3ZanhLk1LVNG6FrgG7a15pKX+t7hzWtjA
uLSqovn4Jr3tOGGyB9nDoRoWxBYqMlluNenqBgNiLD22DlTMD1iD/NCDEOGq5h5v
3v+LD1NV8yrbRF/dx/GWkH3hl8uiiBaZkGqRI09D10CpuIK2lTsrrqcJyMmiG+8n
gqKikDCCBO4GCSqGSIb3DQEHATAdBglghkgBZQMEASoEEMN3AKX128vExYesH/M/
yzSAggTA9ak1ngvEX38Jv3hlx0Jel99rFou3EqBvZw4VvZ7y3B0ZycNu+Yk39XSn
yDrGBZ84K6sIF/n2DJTz8dZfLxy1iTtZRZ+f9zsbAqtKzz2JTLH4fYJSyTnAC3lU
38z6cLVnMyhZliU/zmE0kU+b0CGoI71ubQhJvYtyMraC5Z94/VHkeYyn1fR8YMHU
OCoJLtjK2Kz5VPuSZNLrBLQXSOEzLhR/QhTv8x+/nW6t1WnHjFGgq3yYyNysQgZw
qlkfOuYtEpLEZM3kBxY/Hbb7hhN4g2UIx3IiYxCOC97mxWfM1YbyaHmt2fuZYW+V
JVqiOqHyVyYI6an5z6FsrSfdFN4hSLFowL44ky669i2JlkROQ//CcCV30+gL8VvK
c6BvYRskuvvUDttmzVhsciugvCI5HuP3PLNGFejDqENX2nkJatPQwJv+rDsnHMN6
M0fh+fVbJ3vJosR22QBLF+kopxj6xD725PUQh36GyoHq3V1aT7VtH7HIqR1PMOsn
wZRK+lUT1Jj0Yqv2gkOM4XWMx3vL5ZJ7c7qc95i3uzUhSj8fr6TKkYMcVYQROYJd
GD6EODcw3jmocDx7I4uvGGnb2GP3N8QmNJrBeJnBQCZtmsgiOeFnV1QHvqoFCG/m
+aHrv6a6drK5bOlzK2pelUar6O/XaKcVr7ZWjgFWG6Wbudd3DVBU7muVUCiBbrqJ
G3aT/z+qDK1AcBe2QdUfuk0v+QTa8jDatbypor0bv/wpfSQ81yl40edMyyXv6ZxY
ZKBcZKGfeSn5cF3h1gt0hSrpVZIGscb/Xehx8unBl4bjzfGkaUhu7kFo5WD3fVKr
PKAC8GtCVa3vDFAI5d1B8PFz1DaT1QTQhlHSVmXNpsjIGZujqR1sLGQU+XWyy3qu
gDYZEFcK1BjUhtMG4uVKz2Cm3AVOWZU/EzVpiBnxDLirE9z6YdoXZjhiMCnOpAps
C8UDAqMvxRLYqadJz67qt6yaY7xFLqcihz0uME46midfMbdI94ztkLXt6D70ML15
Q6Q3QbHS8NKgXKJ1NZeo6CGFgagj9OoaJjr400cFz/dAhgDVvE8AAKQTZHUCIvAr
iKy/Y/VS1WySNETNKeUgj4uOpZqwVvhGkQYYeZVjYXrrWlyN6B4pmFXLNl5hoOsM
6zWm5zaKY2gQJzTbHnCqkeqxkhfZeRXpkYqiTT86hzy+AsaXGnQXJcTHROlwrkbU
9gxlduIKOVd0uFbpwlBp+304JsuXOfCwyAWt4y3DmCfO7rJxAr1EoCZL2wRkk+xK
di08gMehw8YD4rERNsxg/5kuX1VevfYBR+94cVpg+u6dJtMM1EWazmnGGxnNvItb
vfDAVEgFkFTRn/aLM7nzMgQkythzJS46S878HJ18plTpRJTARtpW9uqllNwh6LnL
NC1z1eYl5dS/s0ErVOxERwaDKx6x3vxaa5hniW8e+yABgSqunrTdnQoQ0dha2Cpr
uXOmwlJyBuclZSEgsgMVVswn/R8x0pIiVW96YU5H+P59bguP5hLnSFvFhLhDades
bG8sRC7dAW87ZHFOGO315872wVsUw0fjGwgLcF6BJ4CtDM/DD2dhV090225gXVCT
```

Author's Address

Benjamin Deutsch

Email: spreakenze@gmail.com

Full Copyright Statement

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Copyright (c) 2018 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Intellectual Property Statement

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other

documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html