Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: March 23, 2017

P. Hallam-Baker
Comodo Group Inc.
September 19, 2016

Mathematical Mesh: Architecture
draft-hallambaker-mesh-architecture-02

Abstract

The Mathematical Mesh 'The Mesh' is an end-to-end secure infrastructure that facilitates the exchange of configuration and credential data between multiple user devices. The architecture of the Mesh and examples of typical applications are described.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Mathematical Mesh is a user centered Public Key Infrastructure that uses cryptography to make computers easier to use.

The Mesh uses cryptography and an untrusted cloud service to make management of computer configuration data transparent to the end user. Each Mesh user has a personal profile that is unique to them and contains a set of public keys for maintaining the user's Mesh profile.

2. Definitions

2.1. Requirements Language

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].

3. Background

Public Key Cryptography permits Internet applications to be secure but requires an infrastructure for key distribution.

WebPKI has been very successful for E-commerce. Client side PKI has been remarkably less successful.

S/MIME and OpenPGP both have significant user bases but both have been limited to a small community. Government for S/MIME, system admins and security researchers for OpenPGP. Use of PKI for authentication of Web users has seen negligible use.

One of the chief obstacles any network application has to overcome is the critical mass problem. While S/MIME and OpenPGP both have several million users, this is a small fraction of the number of email users.

It is likely that the more significant obstacle to deployment is the difficulty of using client side PKI applications. While S/MIME and

OpenPGP both claim to reduce the effort of sending secure email 'to a single click', no security feature that requires the user to make a conscious decision to use it every time it is used can ever hope to achieve ubiquitous deployment.

Attempting to automate the process of sending encrypted mail introduces a new problem. The fact that a user has configured a client to receive encrypted mail the past does not mean that they are capable of receiving and decrypting such mail today. And even if they are still capable of receiving the encrypted mail today, this capability may be limited to a single machine that they do not currently have access to.

While such objections have been repeatedly dismissed as trivial and 'easily solved' by protocol designers, to ordinary email users, they are anything but trivial. If a change is to be made to an infrastructure they rely on daily, it must be completely transparent. An email security infrastructure that interrupts or disrupts their flow of work is totally unacceptable.

Equally overlooked by application designers is the difficulty of configuring applications that support end-to-end security through cryptography. While working on this project, the author attempted to configure a very popular email client to make use of the built in S/MIME capabilities. Even with 25 years of experience, this took over half an hour and required the user to follow a procedure with 17 different steps!

It is important to note that this complexity is not simply a consequence of one poorly designed application, it is the result of the functions of the PKI being divided across three poorly integrated applications on the user's machine compounded by a set of network protocols that are not designed to provide a seamless user experience.

A similar problem is illustrated by the problem of configuring SSH. There is a simple way to configure SSH and there is a secure way and these are not the same. The simple way to configure SSH is for each user to create a single keypair and copy it to each of the machines they might need terminal access to. While this is straightforward it means that there is no way to mitigate the possibility of the key being compromised if a machine is lost or stolen. Sharing a private key between machines is as bad as sharing a password between accounts. But attempting to achieve cryptographic hygiene across a diverse collection of devices requires user effort proportional to the square of the number of devices.

3.1. What it means to be user-centered

A key principle that guides the design of the Mesh is that any set of instructions that can be written down and given to a user can be written down as code and executed by the computer. Public key cryptography is used to automate the process of managing public keys.

Traditional PKI attempted to solve the problems that were of paramount concern to the designers. The designers of S/MIME were concerned with the problem of exchanging secure email within a hierarchical organization and built a (mostly) hierarchical design. The designers of OpenPGP were concerned with the risk of government subversion of the trust infrastructure for nefarious ends.

But what does the user care about? What is the user's principal concern?

The biggest concern I hear from users is not the risk that someone else might get to see their confidential data, rather it is the risk that they might lose their precious data by some unintended user-error.

Being user centered means considering and addressing the requirements that are set by users regardless of whether they are compatible with the designer's view of optimal security. In particular a user-centered PKI must address requirements such as:

Guaranteeing that data loss does not happen even in the most extreme cases of total loss or destruction of all hardware they used to store their keys.

Mitigating the consequences of user error or carelessness.

Mitigating the consequences of devices being lost or stolen.

Providing mechanisms that permit a user to permit access to their digital assets after their death.

3.2. Eliminate unnecessary options

Traditionally cryptographic applications give the user a bewildering choice of algorithms and options. They can choose to have one RSA keypair used for encryption and signature or they can have separate keys for both, they can encrypt their messages using 3DES or AES at 128, 192 or 256 bit security. And so on.

The Mesh eliminates such choices as unnecessary. Except where required by an application, the Mesh always uses separate keys for

encryption and signature operations and only uses the highest strength on offer. Currently, Mesh profiles are always encrypted using RSA with a 2048 bit key, AES with a 256 bit key and SHA-2-512. (The CFRG ECC curves will be added in the near future when implementations become available.)

For similar reasons, every Mesh master profile has an escrow key. The use of key escrow by applications is optional, but every profile has the capability of using it should circumstances require.

3.3. Why change is possible

All four of the open standards based PKIs that have been developed in the IETF are based on designs that emerged in the mid-1990s. Performing the computations necessary for public key cryptography without noticeable impact on the speed of user interaction was a constraint for even the fastest machines of the day. Consequently, PKI designs attempted to limit the number of cryptographic operations required to the bare minimum necessary. There were long debates over the question of whether certificate chains of more than 3 certificates were acceptable.

Today a 32 bit computer with two processing cores running at 1.2GHz can be bought for $5 and public key algorithms are available that provide a higher level of security for less computation time. In 1995, the idea that a single user might need a hundred public key pairs and a personal PKI to manage them as an extreme scenario. Today when the typical user has a phone, a tablet and a laptop and their home is about to fill up dozens if not hundreds of network connected devices, the need to manage large numbers of keys for individual users is clear.

Almost any information security requirement has a straightforward solution if you are prepared to commit the necessary resources. In general, each degree of cryptographic separation that is required will introduce an additional layer of hierarchy.

Traditionally PKI has focused on the problem of delegating trust from one party to another. Such capabilities have been implicit in the model but only expressed in applications to a limited degree.

In the WebPKI, Certificate Authorities maintain the private keys corresponding to their widely distributed root keys in offline facilities that are never connected to the Internet. These keys are in turn used to sign 'intermediate root certificates' corresponding to the keys used to sign end entity certificates. The CA has this capability but the end entity does not. In the PKIX model it is assumed that if the end entity needs to change their cryptographic

configuration, they will go back to their CA and get a new certificate.

In the OpenPGP Web of trust, Alice signs the key of Bob who signs the key of Carol. Since everyone is a trust provider in the OpenPGP model, Alice can sign a key for Alice. This mechanism is used to support key rollover but the task of distributing her new keys to the devices where Alice needs them is a problem left to Alice.

While it is quite possible for a very capable and experienced PKI expert to configure PKIX and OpenPGP applications in a fashion that supports management of personal keys, such use is far beyond what can reasonably be expected of typical users.

The Mesh applies PKI technology to the problem of making PKI use effortless. Once an initial configuration is established, the user is not required to think about PKI at all. Every PKI operation (e.g. key and certificate rollover) is performed automatically.

4. Basic Concepts

4.1. Parties

The Mesh is a network infrastructure. As with any such infrastructure it is formed not as a set of things but rather as the relationship between those things.

4.1.1. User

A Mesh user is a person or organization that has established a Mesh personal profile. A Mesh personal profile describes the configuration of the set of devices and applications that the user uses. Each Mesh profile is identified by a globally unique fingerprint value.

A Mesh user MAY have multiple profiles for the purpose of compartmentalizing their online identity and preventing activity in one network context being linked to activity in another network context. The extent to which such separation provides increased privacy is not currently understood. From the point of view of the Mesh protocols, such profiles are held by separate users.

At present the Mesh specifications are designed to support requirements arising from personal use such as the user transferring application settings from one device they own to another device they own. To deploy the Mesh in an enterprise environment, features such as the ability to import settings provided by the IT department are highly desirable.

4.1.2. Devices

The Mesh may be used on any computer that has the ability to connect to a network and perform public key cryptography.

Every device that uses the Mesh has a unique device profile that specifies public key pairs that are unique to that device.

When a device is connected to a user's personal profile, it may be an Administration Device or a Connected Device depending on whether it has been assigned an Administration key.

Administration device A device that has access to an administration key for the user's Mesh Personal Profile and is thus authorized to authorize actions such as connecting a new device to the profile, removing devices and creating or removing application profiles.

Connected Device A device that is connected to the Mesh Personal Profile that is not an administration device.

Note that a device MAY be connected to more than one Personal Profile at the same time. For example, an embedded device such as a thermostat might have a single device profile installed during manufacture. If Alice and Bob share the same accommodations where the thermostat is installed, both users might have connected the device to their personal profile.

4.1.3. Portal Provider

Users do not interact with a Mesh Directly. All interaction with the Mesh is mediated by a Portal Provider. The portal provider is responsible for protecting the Mesh from abuse such as Denial of Service attacks, resource exhaustion, spam, etc.

Users interact with a portal provider through an account which has an account identifier in the traditional [RFC5322] format:

<user>@<domain>

Where is an account identifier that is unique to that portal service and is the DNS name of the portal service.

4.1.4. Mesh Provider

4.1.5. InterMesh

4.2. Technology

4.2.1. UDF Fingerprints

The Uniform Data Fingerprint format (UDF) [draft-hallambaker-udf-03] is used to construct names for Mesh data items. UDF employs Base32 [RFC3977] encoding and the SHA-2-512 and SHA-3-512 digest functions to construct fingerprints of varying lengths.

The choice of fingerprint length is a balance between security and compactness of the representation. Longer fingerprints offer higher security but are less convenient. The minimum fingerprint size recommended for use in the Mesh is 25 characters, this presents a work factor of 2^117 to an attacker attempting to generate a signature key matching a particular fingerprint, approximately the same work factor as RSA with 2048 bit keys.

4.2.2. Resolving

In contrast to the URLs resolved by the HTTP protocol which identify a resource by means of a location and a means of retrieval, a UDF fingerprint only identifies a fixed data object and the data type.

A UDF resolution service resolves UDF fingerprints in the same manner that a HTTP server resolves URLs but can only provide a response for the set of fingerprints known to that specific server. Unlike the HTTP service which the client must trust to return the correct resource, every response returned by a UDF resolution service may be validated against the fingerprint presented in the original request. Thus a user of a UDF resolution service is not required to trust it for the integrity of the result received.

4.2.3. Signed Resources

UDF fingerprints provide a probabilistically unique identifier for a static data object but do not provide a direct means of identifying resources that change over time. To identify such resources, digital signatures are used. A public key signature pair is created and the UDF fingerprint of the public key parameters serves as the identifier. The private key is then used to sign either the data object itself or a data object containing a further public key.

The application/pkix-keyinfo content type described in [draft-hallambaker-udf-03] is used to create identifiers for public keys.

4.2.4. Profile

A Mesh profile is a set of configuration settings that is bound to a persistent identifier (a UDF fingerprint).

The Mesh protocols do not put any limit on the size or complexity of Mesh profiles but a Mesh Portal SHOULD impose such limits as are appropriate to avoid abuse such as denial of service attacks.

4.2.5. JSON Encoding

Javascript Object Notation (JSON) [RFC7159] encoding is used to encode all Mesh data objects except for low level cryptographic formats where other encodings are already established.

4.2.6. HTTP Web Service

The Mesh defines two new protocols:

Mesh Portal Protocol (mmm) A client-server protocol that mediates access to a Mesh.

Intermesh Protocol The Intermesh protocol is used to exchange Mesh profile data between portals. It is a flood fill protocol that applies the same principles demonstrated in NNTP [RFC4644].

The DNS SRV mechanism is used for

4.2.7. Transparency

The principle of transparency was introduced by the Certificate Transparency specification [RFC6962]. Transparency is the ability to audit a system using only information that is available to the users of the system. If the system is a public service, all the data used to audit the service must be public.

The Mesh uses strong encryption and

5. Mesh Profiles

5.1. Device Profile

Is unique to each device. If a device has multiple accounts, each account would typically require a separate device profile.

Has separate keys for encryption, authentication and signature.

Typically generated on the device.

Once generated, is typically constant until the device is reset.

Used to provision application keys out to a device.

5.2. Master Profile

Is signed by the Master Signing Key which is in turn validated by the fingerprint.

Contains a Master Signing Key, Set of Administration Keys and Set of Escrow Keys.

Changes infrequently, usually only when the set of administration devices changes or a new escrow key is added.

5.3. Personal Profile

Is signed by an administration key.

For convenience, the master profile is included as an attachment.

Changes when there is a significant change to the configuration, the addition of a new device or application.

5.4. Application Profile

Is signed by an administration key or an application administration key (if specified for the application).

Contains the application configuration data. Is encrypted to the device keys.

Changes when the application configuration is changed or when devices are added or removed.

5.5. Future Directions

It may be desirable to partition the Application profiles so that it is not necessary for every device to download the whole thing. For example, sign a manifest so that the portal can strip out just the parts of the profile that are relevant to a device.

6. Mesh Portal Protocol

Not necessarily instantaneous, may be latency between an update being published and it being available.

7. Intermesh Protocol

This is not a priority at the moment.

May be used to support local replication or replication between providers.

It is anticipated that the Intermesh Protocol will operate at a substantially greater latency than the Mesh Portal Protocol. Probably resynchronizing on an hourly or even daily basis.

Portals are not required to forward every update to the Intermesh. Only updates that have not been superseded within the time quanta need be published.

Each Portal runs a local append only log of every transaction. This is periodically closed and a new log started. Some time after the log is closed, a hash structure is calculated across the log entries and broadcast to the other participants in the InterMesh. After a quorum of hash values has been received, each participant in the exchange calculates a new master hash entry which will be added to the log before the next checkpoint occurs.

The participants exchange log records, but this may be on a limited basis. If the InterMesh has a hundred members, it is not necessary for every single node to have every single entry in real time. It is sufficient for each node to have knowledge of a partner that can provide it on demand.

8. Protocol Overview

[Account request does not specify the portal in the request body, only the HTTP package includes this information. This is probably a bug.]

8.1. Creating a new portal account

A user interacts with a Mesh service through a Mesh portal provider with which she establishes a portal account.

For user convenience, a portal account identifier has the familiar <username>@<domain> format established in [RFC822].

For example Alice selects example.com as her portal provider and chooses the account name alice. Her portal account identifier is alice.

A user MAY establish accounts with multiple portal providers and/or change their portal provider at any time they choose.

8.1.1. Checking Account Identifier for uniqueness

The first step in creating a new account is to check to see if the chosen account identifier is available. This allows a client to validate user input and if necessary warn the user that they need to choose a new account identifier when the data is first entered.

The ValidateRequest message contains the requested account identifier and an optional language parameter to allow the service to provide informative error messages in a language the user understands. The Language field contains a list of ISO language identifier codes in order of preference, most preferred first.

```
POST /.well-known/mmm/HTTP/1.1
Host: example.com
Content-Length: 88

{
  "ValidateRequest": {
    "Account": "alice@example.com",
    "Language": ["en-uk"]}}
```

The ValidateResponse message returns the result of the validation request in the Valid field. Note that even if the value true is returned, a subsequent account creation request MAY still fail.

```
HTTP/1.1 200 OK
Date: Mon 19 Sep 2016 09:00:33
Content-Length: 190

{
  "ValidateResponse": {
    "Status": 201,
    "StatusDescription": "Operation completed successfully",
    "Valid": true,
    "Minimum": 1,
    "InvalidCharacters": ".,:;{}()[]<>?|\\@#"}}
```

[Note that for the sake of concise presentation, the HTTP binding information is omitted from future examples.]

8.2. Creating a new user profile

The first step in creating a new personal profile is to create a Master Profile object. This contains the long term Master Signing Key that will remain constant for the life of the profile, at least one Online Signature Key to be used for administering the personal profile and (optionally), one or more master escrow keys.

For convenience, the descriptions of the Master Signing Key, Online Signing Keys and Escrow Keys typically include PKIX certificates signed by the Master Signing Key. This allows PKIX based applications to make use of PKIX certificate chains to express the same trust relationships described in the Mesh.

```
{
  "MasterProfile": {
    "Identifier": "MDL54-N2LX5-25QIS-TBUVR-B5VN7-D2AXY",
    "MasterSignatureKey": {
      "UDF": "MDL54-N2LX5-25QIS-TBUVR-B5VN7-D2AXY",
      "X509Certificate": "
MIIDJzCCAg-gAwIBAgIRAIfYVuUUSogxe1GiyLqqu1MwDQYJKoZIhvcNAQENBQAw
LjEsMCoGA1UEAxYjTURMNTQtTjJMWDUtMjVRSVMtVEJVVlItQjVWTjctRDJBWFkw
...
VqW8r6bTyjibeFiDyUQGMpB1nqums_NbiYQ0A4Zsu1H-6rrs8y6f6HG6oA"
,
      "PublicParameters": {
        "PublicKeyRSA": {
          "kid": "MDL54-N2LX5-25QIS-TBUVR-B5VN7-D2AXY",
          "n": "
zTjCcO-QvcZjO7L2EF0J6sLYzDuhvOy0Yl0aX-r9-3s61xKGbvYt3rv5NwI44Wdd
Clq9KDjZCPMTTm_gYFopSfUWwvs5GyU8HiG5ZG8VJq7fJy-FdagKEX_9KbEVLff5
EdZH_R28EU47keLItXJ3TTz2SXZwuEsMMVWnhl-T6zzscUm_vsGaBeImuS_9KOoQ
7qEebEkUKlirLKv1CerV8osNaNJsR4eUqZ_SFSyEqtgGFgn3u1O99OgNNiDCRR_q
dxJHatcBepREag9Y3Tp5dz2PhjYsWUGPG0C5Le0L-2goiuwK33Mna3CFYibqU9d2
zB4iKM4EjSucgY-aOoSNIQ"
,
          "e": "
AQAB"
}}},
    "MasterEscrowKeys": [{
        "UDF": "MCIQ4-F7JCR-DIUHN-5QZLR-52BMP-DQBI3",
        "X509Certificate": "
MIIDJjCCAg6gAwIBAgIQT5klM7cMNKPuoQSghOYsgzANBgkqhkiG9w0BAQ0FADAu
MSwwKgYDVQQDFiNNREw1NC1OMkxYNS0yNVFJUy1UQlVWUi1CNVZONy1EMkFYWTAe
...
4G5RLIWE9dszHIa_0BF9C-KqFM4mIfP8txKflSYrHqfhDu0Zp3FL07cT"
,
        "PublicParameters": {
```

```
          "PublicKeyRSA": {
            "kid": "MCIQ4-F7JCR-DIUHN-5QZLR-52BMP-DQBI3",
            "n": "
4CpFv7yGjDj3dS5svPhzTJNUir-B9ArIIVcmZibt8f02i3QFdSIQNNhJ6KI5hgNh
NzuXrktgiAlUJ6N_R3F2uDK9AEJpjwRaiSOAEdSfr7C8Asd-QxwPN6dXYgO7D8cd
2RhBmDFl4NKXxuPeiHexlMMftEdeMyb9QpNnnAN8Rt6_s1a-Ln-pCDTWZCAxDiw4
JVdejUp_sTM2UgcP7uGDUjlz8Pg328O-WBlS_PT95lLXalFE_lVoMJmRXR0p6_kF
rt0es_gqW0n2sZhFb1UjLuq9bRcqn7e9JTmqhxVeD43aPvvyY6V12B3wz1-E3JYH
REAyZe3vJKsWenGlfcdJAw"
,
            "e": "
AQAB"
}}}],
    "OnlineSignatureKeys": [{
        "UDF": "MCY5V-LHCTW-F6HZ6-5PDXQ-HYZYM-YPW6N",
        "X509Certificate": "
MIIDJzCCAg-gAwIBAgIRAOs8crO4N3ufO16_pm2hva4wDQYJKoZIhvcNAQENBQAw
LjEsMCoGA1UEAxYjTURMNTQtTjJMWDUtMjVRSVMtVEJVVlItQjVWTjctRDJBWFkw
...
HHBIJHq324BLTY7vBrQR62QJfMdtFcTfNiroa3RjV57jeLqP6bfqr3Owsg"
,
        "PublicParameters": {
          "PublicKeyRSA": {
            "kid": "MCY5V-LHCTW-F6HZ6-5PDXQ-HYZYM-YPW6N",
            "n": "
-q8C1cRoc2nEv2xM8QChTXD71SQmABJkS0UfBKTv7N6ecxYsZvcfifEo1oY7QuPZ
n6HHZb5qrISUBO3JUcEYpuhKbMYmXXFPCSluTnvRatQxRtK0PsaahUSk29XsNgQF
lqdtUF5G_l9u_Ih6ODRf3B2MkH0Wpw3TKI9ZngtARApBFdZ52IOH9NNWfYeUNZXx
BWzlEuXGfIrtNnA1o7MtCWatrAk-an2BiRtK_mf9bzG3pn1zCI6rRs_NqelOX7pa
b1TFXcIkD5PTRDS0VbN2x8F4pZPihJSnXWtZJwPS3oLC6rFQGUZVChQhiJXe3GTI
GSiDwESeuLGvlNnQX1IrxQ"
,
            "e": "
AQAB"
}}}]}}
```

The Master Profile is always signed using the Master Signing Key:

```
{
  "SignedMasterProfile": {
    "Identifier": "MDL54-N2LX5-25QIS-TBUVR-B5VN7-D2AXY",
    "SignedData": {
      "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTURMNTQtTjJMWDUtMjVRSVMt
VEJVVlItQjVWTjctRDJBWFkifQ"
,
      "payload": "
ewogICJNYXN0ZXJQcm9maWxlIjogewogICAgIklkZW50aWZpZXIiOiAiTURMNTQt
TjJMWDUtMjVRSVMtVEJVVlItQjVWTjctRDJBWFkiLAogICAgIk1hc3RlclNpZ25h
...
IgpBUUFCIn19fV19fQ"
,
      "signature": "
wl8GnlbPI99C1LanNv8220NPltpDW4fjOlCcKj0PDMCrH_rQ3749UXdHv0Dfjrpa
Pvd0gnUL-4_QcPzRcegsZ-buGEgKliabvzr-xTYIbxdJmge9aezvXToWFUVwkWV7
KPgAMApFPEVomf8o80_bhfl8_E1fO9hEca6-8JkDkOKXxZzT8ngLXdqQNcnZcSon
5dZRCJmGdOmCDmADv8gPeNoCCXOK-fJwzsIBY78zcY7nvJDC-ScWeCIZlXmXkSnR
4gCK75tzJlSbAPI7V2Tlm4B5gPOk4fE00TNvCZdRtQS8KKMmbDVP2dtfpARGc9rZ
-620TtBEeYnex5AaU5BJHA"
}}}
```

Since the device used to create the personal profile is typically connected to the profile, a Device profile entry is created for it. This contains a Device Signing Key, a Device Encryption Key and a Device Authentication Key:

```
{
  "DeviceProfile": {
    "Identifier": "MC7SA-FWMVR-WHGHR-2CZUT-TDLDW-TLBDO",
    "Names": ["Alice Desktop"],
    "Description": "A desktop computer built by Acme Computer Co.",
    "DeviceSignatureKey": {
      "UDF": "MC7SA-FWMVR-WHGHR-2CZUT-TDLDW-TLBDO",
      "PublicParameters": {
        "PublicKeyRSA": {
          "kid": "MC7SA-FWMVR-WHGHR-2CZUT-TDLDW-TLBDO",
          "n": "
k-igeP99Z8CiyQm4ZNnPFCutCHKOZm4VNhqmliZKYciSRTMw4bhe7ySN8T3pQgdX
Ib8ii55WaQIxvXlqcYBFfJ5YmrfmoA7OVp96ayaMzY3Ll3sKCCm6a0X11CX9an1Y
FZi_iJlDRLSkH__7Ulgq2IbeZ4GTlhX7i-28s03a9b1HIo-gF8BywJM2ewmACp0s
_JPpRLVpVSpzfgChh3fFchuC1M0u1QuUAR_G7mNgiZJm9cwSLfU5sh9dzm6LBVfR
vXEtrgNaD-dUWPZNcuoq4Iu3hZCOqrtO7MI908c14vQxcjfCTjVwAIzbce1RWXll
U-dgLdR5C4fSkjTrzLmJew"
,
          "e": "
AQAB"
```

```
}}},
    "DeviceAuthenticationKey": {
      "UDF": "MAPO7-SK5VI-PC3NK-MRGWS-HTGBJ-ZRTFA",
      "PublicParameters": {
        "PublicKeyRSA": {
          "kid": "MAPO7-SK5VI-PC3NK-MRGWS-HTGBJ-ZRTFA",
          "n": "
n2DEEO5Xt8GpwHPWSspajsE8PQ2Lol_xhvTdzGJwo_iLvHUbpGz9TD2HdD4QQ8ws
vOn1nAvesWNlWhEWXg1FR8aZ4OA3INOYlPvcJ1spRCJtJLffVk9lCoW4xH92cJEC
eCDGAmedZEo8nUNdpyMk4C1FwtMiX1bxT_FP-0mqG8Z_CL0P30xvG0wmYk2mZBxo
gqR8FhRohuvSp6w0JOdKNqRMCIWg-cXJMfKmg_rohf2g1aunrPujsfy3WAioYFYb
aYsDl4MmQxn8XF8HgrKtCJ5pbVn7WPWZypRx9DznXtixdBqWlmLawt13PmCXcRIl
e0eFfh6D4I5iWo07dhDgiQ"
,
          "e": "
AQAB"
}}},
    "DeviceEncryptiontionKey": {
      "UDF": "MAMNH-KGSPE-RDKMQ-TEKRS-5RRVC-FNIMV",
      "PublicParameters": {
        "PublicKeyRSA": {
          "kid": "MAMNH-KGSPE-RDKMQ-TEKRS-5RRVC-FNIMV",
          "n": "
zYY8R-12ETEcRLsYxpxECAE7o9GDrUMGv4g1W9e7Uw64lc78MldatvAg30xjdKIJ
KYw3PCSfX1c2i-C2lo7Nl8ZGgtnm_SsUvcqh_DIk5risdCnXl0DGTwz8VLDoyg_h
OmyPMDwpMWriZ5DxLDv26VjOF3oP5vMRruz9ooqM8neone6ctKic2EE668XREEwm
jUKz-EyOz4Tm4vjcClihSyOd-vpSaudMcgLhC6A_hlRKdB2lAHKUyyLQpOX4Dae2
kmYn0rwNuF8T3ahk4tbxG0To_azhWQtFdYYg2W1FTIsEKWute9tjkUBq9BWol_RD
WjZrm5fq6JcGtqxxr3E46Q"
,
          "e": "
AQAB"
}}}}}
```

The Device Profile is signed using the Device Signing Key:

```
{
  "SignedDeviceProfile": {
    "Identifier": "MC7SA-FWMVR-WHGHR-2CZUT-TDLDW-TLBDO",
    "SignedData": {
      "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUM3U0EtRldNVlItV0hHSFIt
MkNaVVQtVERMRFctVExCRE8ifQ"
,
      "payload": "
ewogICJEZXZpY2VQcm9maWxlIjogewogICAgIklkZW50aWZpZXIiOiAiTUM3U0Et
RldNVlItV0hHSFItMkNaVVQtVERMRFctVExCRE8iLAogICAgIk5hbWVzIjogWyJB
...
ICAgICAiZSI6ICIKQVFBQiJ9fX19fQ"
,
      "signature": "
ARJOACdfRq9VgHy5gNy3bryFIlmCk4QWjqSMQSIPpXgOzghXYizIk4H4j2loxNNS
jfVEQHB9bwd767RTgnhayVRzI9TfeUDi7GDYpMvpJOl6rUfXXpwECROxyGUnhQfa
eXzfYB6B3dbfYWFqYHHl3_cqjre_sp2EjkZEZ7Y1qiM1U1JQGzdtQrgdOhyXULZY
vSjCVoyCdMIq4it1v5Dri3MxbMwL48B7mBaGKOWPyV-NzFxF0bG4cjgEUOO1YjYG
o-LDehwwVxwu590Y1i6KI3OvUBVobA5BqAHztWXKlswABkGvZhDYH09q_oxGYx3e
h_ObMDoOtZPoYUn-F__Fbw"
}}}
```

A personal profile would typically contain at least one application when first created. For the sake of demonstration, we will do this later.

The personal profile thus consists of the master profile and the device profile:

```
{
  "PersonalProfile": {
    "Identifier": "MDL54-N2LX5-25QIS-TBUVR-B5VN7-D2AXY",
    "SignedMasterProfile": {
      "Identifier": "MDL54-N2LX5-25QIS-TBUVR-B5VN7-D2AXY",
      "SignedData": {
        "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTURMNTQtTjJMWDUtMjVRSVMt
VEJVVlItQjVWTjctRDJBWFkifQ"
,
        "payload": "
ewogICJNYXN0ZXJQcm9maWxlIjogewogICAgIklkZW50aWZpZXIiOiAiTURMNTQt
TjJMWDUtMjVRSVMtVEJVVlItQjVWTjctRDJBWFkiLAogICAgIk1hc3RlclNpZ25h
...
IgpBUUFCIn19fV19fQ"
,
        "signature": "
wl8GnlbPI99C1LanNv8220NPltpDW4fjOlCcKj0PDMCrH_rQ3749UXdHv0Dfjrpa
Pvd0gnUL-4_QcPzRcegsZ-buGEgKliabvzr-xTYIbxdJmge9aezvXToWFUVwkWV7
KPgAMApFPEVomf8o80_bhfl8_E1fO9hEca6-8JkDkOKXxZzT8ngLXdqQNcnZcSon
5dZRCJmGdOmCDmADv8gPeNoCCXOK-fJwzsIBY78zcY7nvJDC-ScWeCIZlXmXkSnR
4gCK75tzJlSbAPI7V2Tlm4B5gPOk4fE00TNvCZdRtQS8KKMmbDVP2dtfpARGc9rZ
-620TtBEeYnex5AaU5BJHA"
}},
    "Devices": [{
        "Identifier": "MC7SA-FWMVR-WHGHR-2CZUT-TDLDW-TLBDO",
        "SignedData": {
          "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUM3U0EtRldNVlItV0hHSFIt
MkNaVVQtVERMRFctVExCRE8ifQ"
,
          "payload": "
ewogICJEZXZpY2VQcm9maWxlIjogewogICAgIklkZW50aWZpZXIiOiAiTUM3U0Et
RldNVlItV0hHSFItMkNaVVQtVERMRFctVExCRE8iLAogICAgIk5hbWVzIjogWyJB
...
ICAgICAiZSI6ICIKQVFBQiJ9fX19fQ"
,
          "signature": "
ARJOACdfRq9VgHy5gNy3bryFIlmCk4QWjqSMQSIPpXgOzghXYizIk4H4j2loxNNS
jfVEQHB9bwd767RTgnhayVRzI9TfeUDi7GDYpMvpJOl6rUfXXpwECROxyGUnhQfa
eXzfYB6B3dbfYWFqYHHl3_cqjre_sp2EjkZEZ7Y1qiM1U1JQGzdtQrgdOhyXULZY
vSjCVoyCdMIq4it1v5Dri3MxbMwL48B7mBaGKOWPyV-NzFxF0bG4cjgEUOO1YjYG
o-LDehwwVxwu590Y1i6KI3OvUBVobA5BqAHztWXKlswABkGvZhDYH09q_oxGYx3e
h_ObMDoOtZPoYUn-F__Fbw"
}}],
    "Applications": []}}
```

The personal profile is then signed using the Online Signing Key:

```
{
  "SignedPersonalProfile": {
    "Identifier": "MDL54-N2LX5-25QIS-TBUVR-B5VN7-D2AXY",
    "SignedData": {
      "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUM3U0EtRldNVlItV0hHSFIt
MkNaVVQtVERMRFctVExCRE8ifQ"
,
      "payload": "
ewogICJQZXJzb25hbFByb2ZpbGUiOiB7CiAgICAiSWRlbnRpZmllciI6ICJNREw1
NC1OMkxYNS0yNVFJUy1UQlVWUi1CNVZONy1EMkFYWSIsCiAgICAiU2lnbmVkTWFz
...
T3RaUG9ZVW4tRl9fRmJ3In19XSwKICAgICJBcHBsaWNhdGlvbnMiOiBbXX19"
,
      "signature": "
MNxXLSp07njavPlAmUNcysBp0JSLFFNQzEBB4RYp_JZue8RThqFqU8424kMtoHI1
HqsQP_QhezA6GFQqJuxxDqv80j0YLE05uYsN7kyVxyeRjD7YensHS5QslaALWTb-
bl8DZBafC9i7lJ6u35pDYdIuvIhCDh8ZsADwMh0-96QzpoTZHJlSh0f6XxaDhYkY
aZvxxOimIfKnAXJ3rAeaj-eo_L5UTJinRMzEJ0ICpMTwskpBRf01cQP46RHcxqAy
NcDBo5-4-gU9CG6A-eD9YbOqFRBoET1v_jw2b2huj_SFZ5oZ4OaHSLaFuNhZs77D
i1A5KqezpwAikbRoGRWICA"
}}}
```

8.2.1. Publishing a new user profile

Once the signed personal profile is created, the client can finaly make the request for the service to create the account. The request object contains the requested account identifier and profile:

```
{
  "CreateRequest": {
    "Account": "alice",
    "Profile": {
      "SignedPersonalProfile": {
        "Identifier": "MDL54-N2LX5-25QIS-TBUVR-B5VN7-D2AXY",
        "SignedData": {
          "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUM3U0EtRldNVlItV0hHSFIt
MkNaVVQtVERMRFctVExCRE8ifQ"
,
          "payload": "
ewogICJQZXJzb25hbFByb2ZpbGUiOiB7CiAgICAiSWRlbnRpZmllciI6ICJNREw1
NC1OMkxYNS0yNVFJUy1UQlVWUi1CNVZONy1EMkFYWSIsCiAgICAiU2lnbmVkTWFz
...
T3RaUG9ZVW4tRl9fRmJ3In19XSwKICAgICJBcHBsaWNhdGlvbnMiOiBbXX19"
,
          "signature": "
MNxXLSp07njavPlAmUNcysBp0JSLFFNQzEBB4RYp_JZue8RThqFqU8424kMtoHI1
HqsQP_QhezA6GFQqJuxxDqv80j0YLE05uYsN7kyVxyeRjD7YensHS5QslaALWTb-
bl8DZBafC9i7lJ6u35pDYdIuvIhCDh8ZsADwMh0-96QzpoTZHJlSh0f6XxaDhYkY
aZvxxOimIfKnAXJ3rAeaj-eo_L5UTJinRMzEJ0ICpMTwskpBRf01cQP46RHcxqAy
NcDBo5-4-gU9CG6A-eD9YbOqFRBoET1v_jw2b2huj_SFZ5oZ4OaHSLaFuNhZs77D
i1A5KqezpwAikbRoGRWICA"
}}}}}
```

The service reports the success (or failure) of the account creation request:

```
{
  "CreateResponse": {
    "Status": 201,
    "StatusDescription": "Operation completed successfully"}}
```

8.3. Connecting a device profile to a user profile

Connecting a device to a profile requires the client on the new device to interact with a client on a device that has administration capabilities, i.e. it has access to an Online Signing Key. Since clients cannot interact directly with other clients, a service is required to mediate the connection. This service is provided by a Mesh portal provider.

All service transactions are initiated by the clients. First the connecting device posts ConnectStart, after which it may poll for the outcome of the connection request using ConnectStatus.

Periodically, the Administration Device polls for a list of pending connection requests using ConnectPending. After posting a request, the administration device posts the result using ConnectComplete:

```
Connecting                  Mesh                 Administration
  Device                   Service                   Device

        |                         |                         |
        |      ConnectStart       |                         |
        | ----------------------> |                         |
        |                         |      ConnectPending     |
        |                         | <---------------------- |
        |                         |                         |
        |                         |      ConnectComplete    |
        |                         | <---------------------- |
        |      ConnectStatus      |                         |
        | ----------------------> |                         |
```

The first step in the process is for the client to generate a device profile. Ideally the device profile is bound to the device in a read-only fashion such that applications running on the device can make use of the deencryption and authentication keys but these private keys cannot be extracted from the device:

```
{
  "DeviceProfile": {
    "Identifier": "MBOER-4FZVP-J3DVQ-ZH4SZ-N4NPG-C7MAI",
    "Names": ["Alice Ring"],
    "Description": "A wearable ring computer bought.",
    "DeviceSignatureKey": {
      "UDF": "MBOER-4FZVP-J3DVQ-ZH4SZ-N4NPG-C7MAI",
      "PublicParameters": {
        "PublicKeyRSA": {
          "kid": "MBOER-4FZVP-J3DVQ-ZH4SZ-N4NPG-C7MAI",
          "n": "
3r_b_iNt7h88KzATdEOH1zR4AHXA-ATjEzmvwFQTb2RErece37Ca5iQKwuuWgcKu
LRxqgplvuDYiuXGDL9DIJs2aHaSf9hZldLWdm-LduCcuF5oFKfNcNCdywD1j6xTa
RgnEd_dmV4Dnwn_ep80wBx-iQgDS1SW0AgoQo2867VNg26kElzi46CX7GGmDSXwB
pIfndEMv6xTm07omUzUueVwFleNRUQ48ESZzAzqF6GR993imAYisPZMFoFXZElXf
KBFZpA57X8g6cAlmzcMayxXwh27K-3v_8S0RfOUlo4GQ8yO2I-X2ix1lTbzopeSO
cJ1HoGojVCRlvbIEXO4PEw"
,
          "e": "
AQAB"
}}},
    "DeviceAuthenticationKey": {
      "UDF": "MCE2J-BZZ6K-63MVK-WRODF-KCJAC-MA424",
      "PublicParameters": {
```

```
        "PublicKeyRSA": {
          "kid": "MCE2J-BZZ6K-63MVK-WRODF-KCJAC-MA424",
          "n": "
uulQYNaIyNPLCaAVH2yQZdeHIdOqurEZCrMoCuTDRFu4Ie3KBB-VyicGSKICp0uI
b2fhXheFdW8vX4f98oO0WR-UjzKszkKAdxi3P3JCsTLAwkcSB4GqzBULmxBt576r
qfLAHYI8QYx5uPZPhABqI02IZefxYZ06jgAWEp44jtOQHe9rARxdQJLbgzVR1H23
E9VDF-10CVBegmTrmK_-Sj8HZV6fAeLtOISVmTEsFZnHobddwkTSOd34DhY0cQg7
Y2ftdKcIwdQCRdTv5_n1NsE7nrKcNngWgULJbfSw0cHOYiE7_Og8Ljy1puWEZMSN
e0UajeSrrkUtw6ZnNcDGxQ"
,
          "e": "
AQAB"
}}},
    "DeviceEncryptiontionKey": {
      "UDF": "MB6UF-V26OG-ME234-PHNWP-NS4RQ-V64YO",
      "PublicParameters": {
        "PublicKeyRSA": {
          "kid": "MB6UF-V26OG-ME234-PHNWP-NS4RQ-V64YO",
          "n": "
8YOoeYiiTUppFMnrYfzmDHGTOhulBKQtKsZ-Pdbwd6NWuwZNt6LAT5XpcatcfWZ0
3mulQPY6Dwk90jNtyfyROOf-7nvPRdQneYzBRScFa02M_aL8Paw9gJQ9dHBZ7p62
SHTzFUHTXQMOK1gzV2QQSHdu-j1qY7osjuNPboJsYczCGa3TJBhfwC6l83EfrsP-
NaiByIUESIRhgnfqoD2XTL-d0GC5HwMMLi6PUnQfWTyOPsoU_xB-JRVllYs-YD9t
jmAXQSqU98GdbNWBxJG7EM-QOvrgPIVYOXQE_wHP3e8GMMGCidc4fMQCkWACNeGi
Pe5Io_ViR17JzoH0voX3DQ"
,
          "e": "
AQAB"
}}}}}
```

The device profile is then signed:

```
{
  "SignedDeviceProfile": {
    "Identifier": "MBOER-4FZVP-J3DVQ-ZH4SZ-N4NPG-C7MAI",
    "SignedData": {
      "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUJPRVItNEZaVlAtSjNEVlEt
Wkg0U1otTjROUEctQzdNQUkifQ"
,
      "payload": "
ewogICJEZXZpY2VQcm9maWxlIjogewogICAgIklkZW50aWZpZXIiOiAiTUJPRVIt
NEZaVlAtSjNEVlEtWkg0U1otTjROUEctQzdNQUkiLAogICAgIk5hbWVzIjogWyJB
...
In19fX19"
,
      "signature": "
NTUVKrhLqpQkd6yoo8Lc7dwMTPDcRKXWfsuOUYbqi3Lm3tl4PSULNm-mv0yHStCb
hD3qXhh89-Tk3cijsU6T56_h8boqlFnSfc_UNxdSAyUxKXkjCs6b8BLWa6NxDu6I
79FoWn4jNt7JB-dBi0IS5SW_4Wl1kzHuEjaDL4hZs17h7TDmHX_gB6iF_p5wR1sT
GqpxINpOW6l9v6lNfEUeQQOU1dvzeK8_3dYQX25rnURA5wnSUxWhRSieGeXvAR8k
7J8IY4jxNaGH8ncSe2g4JNCsrC0PpwqjzVuCye1Mf_Kr35e1JgwzXMTxhURegC_-
ejdf6vcSl9GuFQz57KeZYg"
}}}
```

8.3.1. Profile Authentication

One of the main architecutral principles of the Mesh is bilateral authentication. Every device that is connected to a Mesh profile MUST authenticate the profile it is connecting to and every Mesh profile administrator MUST authenticate devices that are connected.

Having created the necessary profile, the device MUST verify that it is connecting to the correct Mesh profile. The best mechanism for achieving this purpose depends on the capabilities of the device being connected. The administration device obviously requires some means of communicating with the user to serve its function. But the device being connected may have a limited display capability or no user interaction capability at all.

8.3.1.1. Interactive Devices

If the device has user input and display capabilities, it can verify that it is connecting to the correct display by first requesting the user enter the portal account of the profile they wish to connect to, retreiving the profile associated with the device and displaying the profile fingerprint.

The client requests the profile for the requested account name:

```
{
  "GetRequest": {
    "Account": "alice",
    "Multiple": false}}
```

The response contains the requested profile information.

```
{
  "GetResponse": {
    "Status": 201,
    "StatusDescription": "Operation completed successfully",
    "Entries": [{
        "SignedPersonalProfile": {
          "Identifier": "MDL54-N2LX5-25QIS-TBUVR-B5VN7-D2AXY",
          "SignedData": {
            "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUM3U0EtRldNVlItV0hHSFIt
MkNaVVQtVERMRFctVExCRE8ifQ"
,
            "payload": "
ewogICJQZXJzb25hbFByb2ZpbGUiOiB7CiAgICAiSWRlbnRpZmllciI6ICJNREw1
NC1OMkxYNS0yNVFJUy1UQlVWUi1CNVZONy1EMkFYWSIsCiAgICAiU2lnbmVkTWFz
...
T3RaUG9ZVW4tRl9fRmJ3In19XSwKICAgICJBcHBsaWNhdGlvbnMiOiBbXX19"
,
            "signature": "
MNxXLSp07njavPlAmUNcysBp0JSLFFNQzEBB4RYp_JZue8RThqFqU8424kMtoHI1
HqsQP_QhezA6GFQqJuxxDqv80j0YLE05uYsN7kyVxyeRjD7YensHS5QslaALWTb-
bl8DZBafC9i7lJ6u35pDYdIuvIhCDh8ZsADwMh0-96QzpoTZHJlSh0f6XxaDhYkY
aZvxxOimIfKnAXJ3rAeaj-eo_L5UTJinRMzEJ0ICpMTwskpBRf01cQP46RHcxqAy
NcDBo5-4-gU9CG6A-eD9YbOqFRBoET1v_jw2b2huj_SFZ5oZ4OaHSLaFuNhZs77D
i1A5KqezpwAikbRoGRWICA"
}}}]}}
```

Having received the profile data, the user can then verify that the device is attempting to connect to the correct profile by verifying that the fingerprint shown by the device attempting to connect is correct.

8.3.1.2. Constrained Interaction Devices

Connection of an Internet of Things 'IoT' device that does not have the ability to accept user input requires a mechanism by which the user can identify the device they wish to connect to their profile and a mechanism to authenticate the profile to the device.

If the connecting device has a wired communication capability such as a USB port, this MAY be used to effect the device connection using a

standardized interaction profile. But an increasing number of constrained IoT devices are only capable of wireless communication.

Configuration of such devices for the purpose of the Mesh requires that we also consider configuration of the wireless networking capabilities at the same time. The precise mechanism by which this is achieved is therefore outside the scope of this particular document. However prototypes have been built and are being considered that make use of some or all of the following communication techniques:

o

  * Wired serial connection (RS232, RS485).

  * DHCP signalling.

  * Machine readable device identifiers (barcodes, QRCodes).

  * Default device profile installed during manufacture.

  * Optical communication path using camera on administrative device and status light on connecting device to communicate the device identifier, challenge nonce and confirm profile fingerprint.

  * Speech output on audio capable connecting device.

8.3.2. Connection request

After the user verifies the device fingerprint as correct, the client posts a device connection request to the portal:

```
   {
     "ConnectStartRequest": {
       "SignedRequest": {
         "Identifier": "MBOER-4FZVP-J3DVQ-ZH4SZ-N4NPG-C7MAI",
         "SignedData": {
           "header": "
   ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUJPRVItNEZaVlAtSjNEVlEt
   Wkg0U1otTjROUEctQzdNQUkifQ"
   ,
           "payload": "
   ewogICJDb25uZWN0aW9uUmVxdWVzdCI6IHsKICAgICJQYXJlbnRVREYiOiAiYWxp
   Y2UiLAogICAgIkRldmljZSI6IHsKICAgICAgIklkZW50aWZpZXIiOiAiTUJPRVIt
   ...
   WWcifX19fQ"
   ,
           "signature": "
   PHQjSWlP_mCIycmIppZQYkHssweytf1ala7Ypq2D1u7-8GyYH1HcOrkIyZRskt4G
   5X7P352MfSBBSAZT99ME0fTS4-otg8K5Ctn8jBY5IVD6PLB07cF8lGYIPTUpRnJJ
   qVJ2FeGUJqFb8kHd1qF0AfbJL3LbZq3zljhEjgfUHEwzefaX1nwpV2S6muqFC3rQ
   WRJnHd8I9Uoxvr310lQ7PXQC0ZswzOkBSaQQEfntLorZHDnVf4m_cTOsYTbqg9fJ
   EvPVhRAT0Fyw_lYp_Byc-5P9D7A0IqK8feJvGFIxPhbktfmIoLGLP-ooCO9082ln
   oW4OFywAleIcbWEq8sbu0g"
   }},
       "AccountID": "alice"}}
```

The portal verifies that the request is accepable and returns the transaction result:

```
   {
     "ConnectStartResponse": {}}
```

8.3.3. Administrator Polls Pending Connections

The client can poll the portal for the status of pending requests at any time (modulo any service throttling restrictions at the service side). But the request status will only change when an update is posted by an administration device.

Since the user is typically connecting a device to their profile, the next step in connecting the device is to start the administration client. When started, the client polls for pending connection requests using ConnectPendingRequest.

```
   {
     "ConnectPendingRequest": {
       "AccountID": "alice"}}
```

The service responds with a list of pending requests:

```
{
  "ConnectPendingResponse": {
    "Pending": [{
        "Identifier": "MBOER-4FZVP-J3DVQ-ZH4SZ-N4NPG-C7MAI",
        "SignedData": {
          "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUJPRVItNEZaVlAtSjNEVlEt
Wkg0U1otTjROUEctQzdNQUkifQ"
,
          "payload": "
ewogICJDb25uZWN0aW9uUmVxdWVzdCI6IHsKICAgICJQYXJlbnRVREYiOiAiYWxp
Y2UiLAogICAgIkRldmljZSI6IHsKICAgICAgIklkZW50aWZpZXIiOiAiTUJPRVIt
...
WWcifX19fQ"
,
          "signature": "
PHQjSWlP_mCIycmIppZQYkHssweytf1ala7Ypq2D1u7-8GyYH1HcOrkIyZRskt4G
5X7P352MfSBBSAZT99ME0fTS4-otg8K5Ctn8jBY5IVD6PLB07cF8lGYIPTUpRnJJ
qVJ2FeGUJqFb8kHd1qF0AfbJL3LbZq3zljhEjgfUHEwzefaX1nwpV2S6muqFC3rQ
WRJnHd8I9Uoxvr310lQ7PXQC0ZswzOkBSaQQEfntLorZHDnVf4m_cTOsYTbqg9fJ
EvPVhRAT0Fyw_lYp_Byc-5P9D7A0IqK8feJvGFIxPhbktfmIoLGLP-ooCO9082ln
oW4OFywAleIcbWEq8sbu0g"
}}]}}
```

8.3.4. Administrator updates and publishes the personal profile.

The device profile is added to the Personal profile which is then signed by the online signing key. The administration client publishes the updated profile to the Mesh through the portal:

```
{
  "PublishRequest": {
    "Entry": {
      "SignedPersonalProfile": {
        "Identifier": "MDL54-N2LX5-25QIS-TBUVR-B5VN7-D2AXY",
        "SignedData": {
          "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUM3U0EtRldNVlItV0hHSFIt
MkNaVVQtVERMRFctVExCRE8ifQ"
,
          "payload": "
ewogICJQZXJzb25hbFByb2ZpbGUiOiB7CiAgICAiSWRlbnRpZmllciI6ICJNREw1
NC1OMkxYNS0yNVFJUy1UQlVWUi1CNVZONy1EMkFYWSIsCiAgICAiU2lnbmVkTWFz
...
fQ"
,
          "signature": "
iGGQq3WSIt4ktFXXt2taK7YFjiuyAQDVWeD12T5KaC1Vor2SzDWq_VIU8nnCH9pD
V_mzARHOfbOwvSGKSB1eeD0bS4Ttzuc3utz_5DnqQaExP87S2wbbeEAZ4y6LxznR
ZR6XiV0UN_HL_wJ4CmqVtCdhipgBZ2e_Vmnyzr5ZEZX0jg9HbbkB7y6FbRcJdPmU
mR7r6delPUh5NXrSotozdQQpUsPqxbkyMzsfRlVeAif1myeC0colzJflPitKkzX7
47EDyZLhREMvo6DcNBhgyO7E-XALPi4CNeqjGpcvV4ik6SMGUVEeeJYhDHj8vGHr
jpwEFkmFKFeMwHa-aCWaiA"
}}}}}
```

As usual, the service returns the response code:

```
{
  "PublishResponse": {
    "Status": 201,
    "StatusDescription": "Operation completed successfully"}}
```

8.3.5. Administrator posts completion request.

Having accepted the device and connected it to the profile, the administration client creates and signs a connection completion result which is posted to the portal using ConnectCompleteRequest:

```
{
  "ConnectCompleteRequest": {
    "Result": {
      "Identifier": "MBOER-4FZVP-J3DVQ-ZH4SZ-N4NPG-C7MAI",
      "SignedData": {
        "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUM3U0EtRldNVlItV0hHSFIt
MkNaVVQtVERMRFctVExCRE8ifQ"
,
        "payload": "
ewogICJDb25uZWN0aW9uUmVzdWx0IjogewogICAgIkRldmljZSI6IHsKICAgICAg
IklkZW50aWZpZXIiOiAiTUJPRVItNEZaVlAtSjNEVlEtWkg0U1otTjROUEctQzdN
...
dGVkIn19"
,
        "signature": "
Pc633ic76UzfiHsCOakNC35TUGAJG8f3t-Qmgjlb6LNfsWGhuu_L3LoP5Eq2R0QK
Xh3TPZPLZcs4PbWKt21IAoU36pFrmHNO3jhKpZJ3leTYVmFmnaVuX4r_qWHMGvfF
98w-6wAMd_vL5VZ9TQQv_l9FJ_H7E4Fpk35Dee0R_ZGKr8rxh0qoyucTB-BghifM
U7tpHza-OzYyeUu9_doPNW4smV8zQNgARerOR6iimYoyO_riTAroMe8C02HbfLkl
ia8mansWUnKKXUvSNHYsEbCmh91C72HqiJX2UUfD_4XMqDs0-ANubRnWQzkqFGAf
_1waHhp7TFfY51T7IVnUdw"
}},
    "AccountID": "alice"}}
```

Again, the service returns the response code:

```
{
  "ConnectCompleteResponse": {}}
```

8.3.6. Connecting device polls for status update.

As stated previously, the connecting device polls the portal periodically to determine the status of the pending request using ConnectStatusRequest:

```
{
  "ConnectStatusRequest": {
    "AccountID": "alice",
    "DeviceID": "MBOER-4FZVP-J3DVQ-ZH4SZ-N4NPG-C7MAI"}}
```

If the response is that the connection status has not changed, the service MAY return a response that specifies a minimum retry interval. In this case however there is a connection result:

```
{
  "ConnectStatusResponse": {
    "Result": {
      "Identifier": "MBOER-4FZVP-J3DVQ-ZH4SZ-N4NPG-C7MAI",
      "SignedData": {
        "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUM3U0EtRldNVlItV0hHSFIt
MkNaVVQtVERMRFctVExCRE8ifQ"
,
        "payload": "
ewogICJDb25uZWN0aW9uUmVzdWx0IjogewogICAgIkRldmljZSI6IHsKICAgICAg
IklkZW50aWZpZXIiOiAiTUJPRVItNEZaVlAtSjNEVlEtWkg0U1otTjROUEctQzdN
...
dGVkIn19"
,
        "signature": "
Pc633ic76UzfiHsCOakNC35TUGAJG8f3t-Qmgjlb6LNfsWGhuu_L3LoP5Eq2R0QK
Xh3TPZPLZcs4PbWKt21IAoU36pFrmHNO3jhKpZJ3leTYVmFmnaVuX4r_qWHMGvfF
98w-6wAMd_vL5VZ9TQQv_l9FJ_H7E4Fpk35Dee0R_ZGKr8rxh0qoyucTB-BghifM
U7tpHza-OzYyeUu9_doPNW4smV8zQNgARerOR6iimYoyO_riTAroMe8C02HbfLkl
ia8mansWUnKKXUvSNHYsEbCmh91C72HqiJX2UUfD_4XMqDs0-ANubRnWQzkqFGAf
_1waHhp7TFfY51T7IVnUdw"
}}}}
```

[Should probably unpack further.]

8.4. Adding an application profile to a user profile

Application profiles are published separately from the personal profile to which they are linked. This allows a device to be given administration capability for a particular application without granting administration capability for the profile itself and the ability to connect additional profiles and devices.

Another advantage of this separation is that an application profile might be managed by a separate party. In an enterprise, the application profile for a user's corporate email account could be managed by the corporate IT department.

A user MAY have multiple application profiles for the same application. If a user has three email accounts, they would have three email application profiles, one for each account.

In this example, the user has requested a PaswordProfile to be created. When populated, this records the usernames and passwords for the various Web sites that the user has created accounts at and has requested the Web browser store in the Mesh.

Unlike a traditional password management service, the data stored the Password Profile is encrypted end to end and can only be decrypted by the devices that hold a decryption key.

```
{
  "PasswordProfile": {
    "Identifier": "MBSHO-5D2GR-T7TNK-XHJV6-BHX5I-KA6TX-A",
    "EncryptedData": {
      "protected": "
ewogICJhbGciOiAiQUUxMjgifQ",
      "iv": "
9LOpFHTVI9-wtPECFHpRoA",
      "ciphertext": "
-jwZ9rngzGV2Gu7x-IwDXGksDFwIm01TJqSPkR_5CacxMB-r0MzVyeomjMmpCImg",
      "recipients": [{
          "Header": {
            "kid": "MAMNH-KGSPE-RDKMQ-TEKRS-5RRVC-FNIMV"},
          "encrypted_key": "
Y5glmB3szauorgusT293C7CAaHMcHtn1PqtmRz7OVcmjz39-xItg0xzhC6-VPXhS
1ye9qJSwmNHv8sQQ8vNi7pWijM-NKFUJhkjEo5iM6CKcpxlphh4gW8exEH-HScvB
5TNZk0wYmmR9LdTribgNlWAwu-n3I9xN9tZAr9icaggGH7GbSO9C9l8IgjpbZsDh
B7bBAuOGNviOy-3mIIF0bXo3EBCS3a_TGFtMicEtWa0AdgpM8hDdOCLZ738HBuxN
vku5SWIGCdx29povPuSalkBuPBX5CFnkq2IqR51QUTRyJtkEl6ts9oprPcJ4GEZd
fgASNb4g4LEP8JC7WmJeQg"},
        {
          "Header": {
            "kid": "MB6UF-V26OG-ME234-PHNWP-NS4RQ-V64YO"},
          "encrypted_key": "
fTFLdjuIw-8uFFmwm5twpiK05F6_tZMO6ZrDjVnjsi_VfOoCPP8hFFemfvx9XqSg
KmhVMysq_sJ6tw40duUNkOZcgaXpkfbBPFlobQPLMIfQydklq_njTTFYHWQhT-ui
aRTu4vUZ_n_R478fF7gKEn5LSCFIDH52AZWgpnRfMsIGdrKJUhPwbQV4vuwd4FFK
uRwLlTMVPfGOUSww38VZx0uIAJdqHztl5gKP3rP2Ld1lpDcGxX-ZYwDQQCdSlH0j
-Z8wezlnbLtTV5ri3TrYpZYEto_TYDVQlPWNArmQhjhrX8q9-pyAxBaEMFA4MhbP
BMN4sq6M34sC6Yc5nS3s1Q"}]}}}
```

The application profile is published to the Mesh in the same way as any other profile update, via a a Publish transaction:

```
{
  "PublishRequest": {
    "Entry": {
      "SignedApplicationProfile": {
        "Identifier": "MBSHO-5D2GR-T7TNK-XHJV6-BHX5I-KA6TX-A",
        "SignedData": {
          "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUM3U0EtRldNVlItV0hHSFIt
MkNaVVQtVERMRFctVExCRE8ifQ"
,
          "payload": "
ewogICJQYXNzd29yZFByb2ZpbGUiOiB7CiAgICAiSWRlbnRpZmllciI6ICJNQlNI
Ty01RDJHUi1UN1ROSy1YSEpWNi1CSFg1SS1LQTZUWC1BIiwKICAgICJFbmNyeXB0
...
QXhCYUVNRkE0TWhiUApCTU40c3E2TTM0c0M2WWM1blMzczFRIn1dfX19"
,
          "signature": "
dOPSeM2MsLRE50l7rcStAAMbqOk2i-AQizwcK9EtAkeCKiYYyspnoD2WEzHDd9K9
mYaKlDvHr4YZ5fPF-tCyqHgNeMO0WjVuaCwSykqwZ-VvQ0IklSwiwTT9IHUYf8S8
WlFYgQOZLfCSropvsbbcBaAH5vZb_OgxXmoZtSzFdn89LQ1W7OiLvj8WLaMZNTcf
x2ChJ0lb3uVs59oUe3NIpaRHjsRvuDIjijN8ga-5tkwOwyfCk7W1u22n9GLEDTSH
DchYRIUUAjd8dxhpyOk_cxcEg263diQvlMGQ7tketBZqdIQZnRZISY5HCAsglgNo
sb-mV33TGfAcf9oOTZp1vw"
}}}}}
```

The service returns a status response.

```
{
  "PublishResponse": {
    "Status": 201,
    "StatusDescription": "Operation completed successfully"}}
```

Note that the degree of verification to be performed by the service when an application profile is published is an open question.

Having created the application profile, the administration client adds it to the personal profile and publishes it:

```
{
  "PublishRequest": {
    "Entry": {
      "SignedPersonalProfile": {
        "Identifier": "MDL54-N2LX5-25QIS-TBUVR-B5VN7-D2AXY",
        "SignedData": {
          "header": "
ewogICJhbGciOiAiUlM1MTIiLAogICJraWQiOiAiTUM3U0EtRldNVlItV0hHSFIt
MkNaVVQtVERMRFctVExCRE8ifQ"
,
          "payload": "
ewogICJQZXJzb25hbFByb2ZpbGUiOiB7CiAgICAiSWRlbnRpZmllciI6ICJNREw1
NC1OMkxYNS0yNVFJUy1UQlVWUi1CNVZONy1EMkFYWSIsCiAgICAiU2lnbmVkTWFz
...
fQ"
,
          "signature": "
iGGQq3WSIt4ktFXXt2taK7YFjiuyAQDVWeD12T5KaC1Vor2SzDWq_VIU8nnCH9pD
V_mzARHOfbOwvSGKSB1eeD0bS4Ttzuc3utz_5DnqQaExP87S2wbbeEAZ4y6LxznR
ZR6XiV0UN_HL_wJ4CmqVtCdhipgBZ2e_Vmnyzr5ZEZX0jg9HbbkB7y6FbRcJdPmU
mR7r6delPUh5NXrSotozdQQpUsPqxbkyMzsfRlVeAif1myeC0colzJflPitKkzX7
47EDyZLhREMvo6DcNBhgyO7E-XALPi4CNeqjGpcvV4ik6SMGUVEeeJYhDHj8vGHr
jpwEFkmFKFeMwHa-aCWaiA"
}}}}}
```

Note that if the publication was to happen in the reverse order, with the personal profile being published before the application profile, the personal profile might be rejected by the portal for inconsistency as it links to a non existent application profile. Though the value of such a check is debatable. It might well be preferable to not make such checks as it permits an application profile to have a degree of anonymity.

```
{
  "PublishResponse": {
    "Status": 201,
    "StatusDescription": "Operation completed successfully"}}
```

8.5. Creating a recovery profile

The Mesh invites users to put all their data eggs in one cryptographic basket. If the private keys in their master profile are lost, they could lose all their digital assets.

The debate over the desirability of key escrow is a complex one. Not least because voluntary key escrow by the user to protect the user's digital assets is frequently conflated with mechanisms to support 'Lawful Access' through government managed backdoors.

Accidents happen and so do disasters. For most users and most applications, data loss is a much more important concern than data disclosure. The option of using a robust key recovery mechanism is therefore essential for use of strong cryptography is to become ubiquitous.

There are of course circumstances in which some users may prefer to risk losing some of their data rather than risk disclosure. Since any key recovery infrastructure necessarily introduces the risk of coercion, the choice of whether to use key recovery or not is left to the user to decide.

The Mesh permits users to escrow their private keys in the Mesh itself in an OfflineEscrowEntry. Such entries are encrypted using the strongest degree of encryption available under a symmetric key. The symmetric key is then in turn split using Shamir secret sharing using an n of m threshold scheme.

The OfflineEscrowEntry identifier is a UDF fingerprint of the symmetric key used to encrypt the data. This guarantees that a party that has the decryption key has the ability to locate the corresponding Escrow entry.

The OfflineEscrowEntry is published using the usual Publish transaction:

```
{
  "PublishRequest": {
    "Entry": {
      "OfflineEscrowEntry": {
        "Identifier": "MBOB3-WIZNV-MH3UA-OQLEI-L44SF-HFGEA",
        "EncryptedData": {
          "protected": "
ewogICJhbGciOiAiQUUxMjgifQ"
,
          "iv": "
1TsjnS6BN3rPH_i4xfh5jQ"
,
          "ciphertext": "
S0WBeEvG0zT8PWQfj3ls_HRt1av-mjYtpem1Q04IdvXCdWMyfAxk5fEUv-TXP-DU
GXlNi70UHLC9dtiAcFQiUcrCjQVJIb6qW6GtzdHrYcZGw8ibgcWKnAFbxk6VK-il
...
9oMj00zipIDIz156EtRo7tl3WTzy6lSJn6aHLjRHYCT6FOyBmSsoamQyP0yE3EyR"
}}}}}
```

The response indicates success or failure:

```
{
  "PublishResponse": {
    "Status": 201,
    "StatusDescription": "Operation completed successfully"}}
```

8.6. Recovering a profile

To recover a profile, the user MUST supply the necessary number of secret shares. These are then used to calculate the UDF fingerprint to use as the locator in a Get transaction:

```
{
  "GetRequest": {
    "Identifier": "MBOB3-WIZNV-MH3UA-OQLEI-L44SF-HFGEA",
    "Multiple": false}}
```

If the transaction succeeds, GetResponse is returned with the requested data.

```
{
  "GetResponse": {
    "Status": 201,
    "StatusDescription": "Operation completed successfully",
    "Entries": [{
        "OfflineEscrowEntry": {
          "Identifier": "MBOB3-WIZNV-MH3UA-OQLEI-L44SF-HFGEA",
          "EncryptedData": {
            "protected": "
ewogICJhbGciOiAiQUUxMjgifQ"
,
            "iv": "
1TsjnS6BN3rPH_i4xfh5jQ"
,
            "ciphertext": "
S0WBeEvG0zT8PWQfj3ls_HRt1av-mjYtpem1Q04IdvXCdWMyfAxk5fEUv-TXP-DU
GXlNi70UHLC9dtiAcFQiUcrCjQVJIb6qW6GtzdHrYcZGw8ibgcWKnAFbxk6VK-il
...
9oMj00zipIDIz156EtRo7tl3WTzy6lSJn6aHLjRHYCT6FOyBmSsoamQyP0yE3EyR"
}}}]}}
```

The client can now decrypt the OfflineEscrowEntry to recover the private key(s).

9. Transparent Audit

Can be performed by any party that is a participant in the InterMesh protocol or subsequently in an offline transaction.

10. Security Considerations

Security Considerations are addressed in the companion document [draft-hallambaker-mesh-reference-02]

11. IANA Considerations

IANA Considerations are addressed in the companion document [draft-hallambaker-mesh-reference-02]

12. Acknowledgements

Comodo Group: Egemen Tas, Melhi Abdulhayo?lu, Rob Stradling, Robin Alden.

13. References

13.1. Normative References

[draft-hallambaker-mesh-reference-02]
"[Reference Not Found!]".

[draft-hallambaker-udf-03]
"[Reference Not Found!]".

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997.

[RFC3977] Feather, C., "Network News Transfer Protocol (NNTP)", RFC 3977, DOI 10.17487/RFC3977, October 2006.

[RFC5322] Resnick, P., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008.

[RFC6962] Laurie, B., Langley, A., and E. Kasper, "Certificate Transparency", RFC 6962, DOI 10.17487/RFC6962, June 2013.

[RFC7159] Bray, T., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014.

13.2. Informative References

[RFC4644] Vinocur, J. and K. Murchison, "Network News Transfer Protocol (NNTP) Extension for Streaming Feeds", RFC 4644, DOI 10.17487/RFC4644, October 2006.

[RFC822]   "[Reference Not Found!]".

Author's Address

Phillip Hallam-Baker
Comodo Group Inc.

Email: philliph@comodo.com