Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 17, 2016

P. Hallam-Baker
Comodo Group Inc.

January 14, 2016

# Mathematical Mesh: Architecture
# draft-hallambaker-mesh-architecture-00

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

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). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on July 17, 2016.

## 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 [RFC5822] 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] 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] 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. Use Scenario

### 5.1. Initial Configuration

### 5.2. Adding a Device

### 5.3. Adding and Updating Applications

### 5.4. Disaster Recovery

## 6. Mesh Profiles

### 6.1. 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.

### 6.2. 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.

### 6.3. 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.

### 6.4. 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.

### 6.5. 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.

#### 6.5.1. Public Profile

#### 6.5.2. Endorsement Statements

## 7. Mesh Portal Protocol

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

## 8. 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.

## 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-architecture]

## 11. IANA Considerations

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

## 12. Acknowledgements

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

## 13. References

### 13.1. Normative References

[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.

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

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

[RFC5822] "[Reference Not Found!]".

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

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

### 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.

Author's Address

Phillip Hallam-Baker
Comodo Group Inc.

Email: philliph@comodo.com