Mathematical Mesh Part I: Architecture Guide

Mesh Architecture

<series>draft-hallambaker-mesh-architecture

<status>informational

<stream>independent

<ipr>trust200902

<author>Phillip Hallam-Baker

<surname>Hallam-Baker

<initials>P. M.

<firstname>Phillip

<email>phill@hallambaker.com

<also>http://mathmesh.com/Documents/draft-hallambaker-mesh-architecture.html

The Mathematical Mesh ‘The Mesh’ is an end-to-end secure infrastructure that makes computers easier to use by making them more secure. The Mesh provides a set of protocol and cryptographic building blocks that enable encrypted data stored in the cloud to be accessed, managed and exchanged between users with the same or better ease of use than traditional approaches which leave the data vulnerable to attack.

The core data structures

This document describes the requirements and constraints that motivate the design of the Mesh and presents the high-level architecture with examples of use.

# Introduction

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

For several decades, it has been widely observed that most users are either unwilling or unable to make even the slightest efforts to protect their security, still less those of other parties. Yet despite this observation being widespread, the efforts of the IT security community has largely focused on changing this user behavior rather that designing applications that respect it.

The Mesh is based on the principle that any effort that is asked of the user is probably more than they are willing to make. If the Internet is to be secure, using applications securely must be effortless. Rather than beginning the design process by imagining all the possible modes of attack and working out how to address these with the least possible inconvenience, we must reverse the question and ask how much security can be provided without requiring any effort on the user's part whatsoever.

## Mesh Naming

A secure communications system requires a secure means of addressing resources.

Secure Internet Names bind an Internet name to a Security Policy by means of a cryptographic digest function. This binding enables an Internet client to determine that an email sent to an address must be end-to-end encrypted under a particular public key or that access to a Web Service requires a particular set of security enhancements.

Content digests are expressed as Uniform Data Fingerprints.

[example SIN]

## Mesh Cryptography

Existing Internet security protocols are based on approaches developed in the 1990s when performance tradeoffs were a prime consideration in the design of cryptographic protocols.

The design of the Mesh is based on the view that a modern security protocol need not consider the capabilities of any machine older than ten years as a constraint on the use of cryptographic security enhancements. Rather than considering whether to apply cryptographic enhancements at the transport, message or data level, the Mesh applies security at all three.

The power of public key encryption comes from the fact that use of separate keys for different roles allows a party to perform one role without being able to perform the other. Mallet can encrypt with Alice's public key but cannot decrypt without her private key. Mallet can verify but not sign.

Existing Internet security protocols only apply public key cryptography with two keys, thus limiting these applications to two roles. The Mesh uses distributed key generation techniques to split decryption keys into two or more parts, thus enabling applications in which there are three or more roles.

One of the most important of these roles is the 'cloud'. Splitting a decryption key into two parts means that both parts must be used to complete a public key transaction. This in turn enables capabilities such as the ability to immediately disable use of a lost or stolen device or true end-to-end secure mailing lists without the need for a static membership list distributed to all the members of the list before a message is sent.

One of the core objectives of the Mesh is to make data level encryption ubiquitous. Recognizing that the worst thing, pictures five years, personal key escrow, does not require secure hardware.

## DARE Messages and Containers

The Data At Rest Encryption (DARE) format is used as the basis for all cryptographic enhancements.

Apply JSON data model and either JSON encoding or JSON-B, a binary extension thereof.

A DARE Message contains a single payload which may be encrypted and/or authenticated.

A DARE Container contains a sequence of DARE Messages typically written as an append-only log. The DARE Container frame format enables efficient traversal of container frames in either the forward or the reverse direction. Container frames may by authenticated individually in the same manner as DARE Messages or as an ensemble using either chained digests or a Merkle Tree.

## Mesh Profiles

Mesh Profile - describes a set of cryptographic credentials

Master profile, a lifelong cryptographic identity. Once created, this should never need to be changed.

Offline management capabilities

Device profile

Account Profile

## Mesh Protocol

Two data structures, set and list

Catalogs, a set of items that is managed by the user. Entries may be added, updated or removed.

Spools, a list of messages that are either inbound or outbound. Messages may be added, marked as read, unread or deleted.

Synchronizing the catalogs and spools associated with an account is sufficient to implement the capabilities of most Internet application protocols.

Besides providing the basis for the Mesh application protocols, the Mesh Catalog Archive provides a possible basis for future distribution and update of the Mesh application software.

## Document Roadmap

I. Architecture

II. Uniform Data Fingerprint

<norm="draft-hallambaker-mesh-udf"/>.

III. Data at Rest Encryption

<norm="draft-hallambaker-mesh-dare"/>.

IV. Schema Reference

<norm="draft-hallambaker-mesh-schema"/>.

V Protocol Reference

<norm="draft-hallambaker-mesh-protocol"/>.

VI The Trust Mesh

<norm="draft-hallambaker-mesh-trust"/>.

VII Security Considerations

<norm="draft-hallambaker-mesh-security"/>.

The following documents describe technologies that are used in the Mesh but do not form part of the Mesh standards suite:

JSON-BCD Encoding

<norm="draft-hallambaker-jsonbcd"/>.

DNS Web Service Discovery

<norm="draft-hallambaker-web-service-discovery"/>.

The following documents describe aspects of the Mesh Reference implementation:

Mesh Developer

<norm="draft-hallambaker-mesh-developer"/>.

Mesh Platform

<norm="draft-hallambaker-mesh-platform"/>.

# Definitions

This section presents the related specifications and standards on which the Mesh is built, the terms that are used as terms of art within the Mesh protocols and applications and the terms used as requirements language.

## Related Specifications

Besides the documents that form the Mesh core, the Mesh makes use of many existing Internet standards, including:

Cryptographic Algorithms

Mesh applications use the cryptographic algorithm suites specified by the application. The cryptographic algorithms used in the Mesh itself are limited to SHA-2 <norm="SHA-2"/> and SHA-3 <norm="SHA-3"/> digest functions, AES Encryption <norm="FIPS197"/> and RSA Signature, and Encryption <norm="RFC8017"/>.

The use of the Ed25519 and Ed448 algorithms is currently being explored for use with *both* signature <norm="RFC8032"/> *and* encryption. The Edwards Curve is preferred over the Montgomery for Encryption as it affords a more straightforward implementation of techniques such as co-generation of public key pairs and proxy re-encryption.

Transport

All Mesh Services make use of multiple layers of security. Protection against traffic analysis and metadata attacks are provided by use of Transport Layer Security <norm="RFC5246"/>. At present, the HTTP/1.1 <norm="RFC7231"/> protocol is used to provide framing of transaction messages.

Encoding

All Mesh protocols and data structures are expressed in the JSON data model and all Mesh applications accept data in standard JSON encoding <norm="RFC7159"/>. The JOSE Signature <norm="RFC7515"/> and Encryption <norm="RFC7516"/> standards are used as the basis for object signing and encryption.

## Defined Terms

TBS

## 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 <norm="RFC2119"/>.

## Implementation Status

The implementation status of the reference code base is described in the companion document <info="draft-hallambaker-mesh-developer"/>.

# User Experience

This section describes the Mesh in use. These use cases described here are re-visited in the companion Mesh Schema Reference <norm="draft-hallambaker-mesh-schema"/> and Mesh Protocol Reference <norm="draft-hallambaker-mesh-protocol"/> with additional details and protocol layer examples.

For clarity and for compactness, these use cases are illustrated using the command line tool meshman. A GUI Mesh management application is likely to be preferable in almost any circumstance other than integration of Mesh features into other tools.

The original design brief for the Mesh was to make it easier to use the Internet securely. Over time, it was realized that users are almost never prepared to sacrifice usability or convenience for security. It is therefore insufficient to minimize the cost of security, if secure applications are to be used securely they must be at least as easy to use as those they replace. If security features are to be used, they must not require the user to make any additional effort whatsoever.

The key to meeting this extreme design constraint is the realization that any set of instructions that is written down and given to a user can be turned into code and executed by machine. Provided that the necessary authentication, integrity and confidentiality controls are provided. Thus the Mesh is not just a cryptographic infrastructure that makes use of computer systems more secure, it is a usability infrastructure that makes computers easier to use by providing security.

The user experience is thus at the heart of the design of the Mesh and a description of the Mesh Architecture properly begins with consideration of the view of the system that matters most: that of the user.

The principle security protocols in use today were designed at a time when most Internet users made use of either a single machine or one of a number of shared machines connected to a shared file store. The problem of transferring cryptographic keys and configuration data between machines was rarely considered and when it was considered was usually implemented badly. Today the typical user owns or makes use of multiple devices they recognize as a computer (laptop, tablet) and an even greater number of devices that they do not recognize as computers but are (almost any device with a display).

[Diagram - Master profile with connected device profiles and service profile]

## Creating and Registering a Mesh Profile

The first step in using the Mesh is to create a personal profile. From the user's point of view a profile is a collection of all the configuration data for all the Mesh enabled devices and services that they interact with.

<include=..\Examples\ArchitectureCreate.md>

Note that the user does not specify the cryptographic algorithms to use. Choice of cryptographic algorithm is primarily the concern of the protocol designer, not the user. The only circumstance in which users would normally be involved in algorithm selection is when there is a transition in progress from one algorithm suite to another.

## Mesh Service

Is chosen by the user.

Can be changed at any time.

Provides a post-office function allowing the user’s devices to exchange messages without requiring direct communication.

Supports a four corner communication model in which all messages pass through inbound and outbound intermediaries.

Mesh Messaging is intentionally limited to short (64KB) messages. This has proved more than sufficient for the longest control messages leaving room for modest sized data messages (contact requests, chat, etc.)

Rather than attempting to support longer messages in the Mesh messaging scheme it is considered preferable to consider direct exchange of very large data sets (Terabytes) as separate ‘detachments’. This model allows such data sets to be transferred by means of a pull model rather than the push-only approach of SMTP.

## Mesh Catalogs

A Mesh Catalog contains a set of entries, each of which has a unique object identifier.

Objects may be added, updated or deleted.

For example, the entries in the credentials catalog specify the users username and password credentials for accessing an Internet service:

<include=..\Examples\ArchitectureCredential.md>

Devices

Contacts

Bookmarks

Credentials

Calendar

By default, all catalog entries are encrypted. Applying the Default Deny principle, in normal circumstances, the Mesh Service is not capable of decrypting any catalog excepting the Device and Contacts catalogs which have special roles as described below.

The Devices catalog is used to assign and remove application authorizations to devices. Devices are authorized by creating and provisioning the necessary decryption keys to the device entry.

The Contacts catalog contains the set of the user’s contacts and is used to apply access control to inbound message requests from third parties.

## Connecting and Authorizing Additional Devices

The primary purpose of the Mesh is to manage user’s cryptographic credentials across multiple devices.

Having established a Mesh profile, a user may connect any number of devices to it.

Devices may be computing devices (laptops, desktops, smart phones), wearable devices, IoT devices.

All devices are connected by means of a connection mechanism that provides for strong mutual authentication. Since the capabilities of devices vary, three connection mechanisms are currently defined: Direct, PIN and QR.

In each case, connection requests are approved by an administration device having access to an administration key authorized by the current Master profile. Administration devices must have data entry (e.g. keyboard) and output (e.g. display) affordances to support any of the connection mechanisms. The QR code connection mechanism additionally requires a suitable camera.

It will be noted that the process of connecting a device that contains a preconfigured set of device keys might in principle expose the user to the risk that the manufacturer has retained knowledge of these keys and that this might be used to effect a ‘backdoor’. The means of controlling this risk by key co-generation is described in a later section.

### Direct Connection

The direct connection mechanism requires that both the administration device and the device originating the connection request have data entry and output affordances and that it is possible for the user to compare the authentication codes presented by the two devices to check that they are identical.

<include=..\Examples\ArchitectureConnectDirect.md>

### Pin Connection

The PIN Connection mechanism is similar to the Direct connection mechanism except that the process is initiated on an administration device by requesting assignment of a new authentication PIN. The PIN is then input to the connecting device to authenticate the request.

<include=..\Examples\ArchitectureConnectPIN.md>

If the Device Profile fingerprint is known at the time the PIN is generated, this can be bound to permit connection of a single device. This mode could be employed

### EARL/QR Code Connection

The EARL/QR code connection mechanisms are used to connect a constrained device to a Mesh profile by means of an Encrypted Authenticated Resource Locator, typically presented as a QR code on the device itself or its packaging.

[image QR Code]

<include=..\Examples\ArchitectureConnectQR.md>

## Contact Requests

As previously stated, every inbound Mesh message is subject to access control. The user’s contact catalog is used as part of the access control authentication and authorization mechanism.

By default, the only form of inbound message that is accepted is a contact request. Though for certain Mesh users (e.g. politicians, celebrities) even contact requests might require some form of prior approval (e.g. endorsement by a mutual friend).

### Remote

In the most general case, the participants are remote and one user must make a contact request of the other.

<include=..\Examples\ArchitectureContactRequest.md>

### Static QR Code

A DARE contact entry may be exchanged by means of an EARL UDF. This is typically presented by means of a QR code.

<include=..\Examples\ArchitectureContactQR.md>

### Dynamic QR Code

If it is possible for the device to generate a new QR code for the contact request, mutual authentication of the credential exchange is possible

<include=..\Examples\ArchitectureContactQR2.md>

## Storing and Sharing Data in the Could

### EARL Exchange

An EARL is a form of URI that specifies a means of locating and decrypting a DARE Message stored on a Web Service.

[Diagram: EARL]

<include=..\Examples\ArchitectureConnectEARL.md>

## Recryption Groups

[Diagram: Recryption]

<include=..\Examples\ArchitectureRecrypt.md>

## Escrow and Recovery of Keys

Availability is the most critical concern

Escrow is optional but recommended

Mesh tools MUST support the escrow and recovery features

Master keys encrypted under symmetric key to create escrow record

Symmetric key split

<include=..\Examples\ArchitectureEscrow.md>

<include=..\Examples\ArchitectureRecovery.md>

# Architecture

## Naming

<include=..\Examples\ArchitectureUDFTypes.md>

### Content Digest

<include=..\Examples\ArchitectureSHA23.md>

### EARL

<include=..\Examples\ArchitectureEARL.md>

## Encoding

### JSON, JSON-B

### Messages

### Containers

## Data Model

### Objects

Unique identifier

Append only log

Log can be purged.

### Catalogs

Set of entries

Entry state machine (Add-Update\*-Delete)\*

### Spools

Queue of messages

Message state machine (Post-(Read-Unread)\*-Delete)

## Catalog Protocol

### Status Transaction

Obtain updated device profile (if it exists) and the status of the set of catalogs the device is authorized

### Download Transaction

Read objects from a catalog or spool owned by the client making the request.

Optional filtering criteria MAY be specified to only return objects matching specific criteria and/or only return certain parts of the selected messages.

The transaction MAY be performed in one request/response round trip or with separate round trips to confirm that the transaction is accepted by the service before sending large volumes of data.

### Upload Transaction

Upload objects to a catalog or spool owned by Read objects from a catalog or spool owned by the client making the request.

Multiple objects MAY be uploaded at once. Object updates MAY be conditional on the successful completion of other upload requests.

The transaction MAY be performed in one request/response round trip or with separate round trips to confirm that the transaction is accepted by the service before sending large volumes of data.

## Spool Protocol

Four corner model enforced.

[Diagram Four corner]

Messages are limited to control with very small amounts of data

Long messages are exchanged as detachments using separate protocol (HTTP).

This ensures that messages are processed quickly and reliably. A server will not be blocked by receipt of a long message.

Aggressive controls on services may be enforced to prevent DoS attacks.

Post transaction is used for client-service and service-service transactions.

### Client-Service (Post Transaction)

### Service-Service (Post Transaction)

### Service-Client (Synchronization)

## User Experience

Sync

* Status
* Upload outbound messages
* Download catalog and spool updates
* Upload catalog updates

Rapid access - only take last 100 messages

Limitation on message size ensures that

# Mesh Data

## Profiles

Describe a set of cryptographic credentials

Usually public keys

### Master Profile

Master Signature key

Master Encryption key

### Device Profile

Signature / Encryption / Authentication

### Mesh Profile

Encryption

## Catalogs

Collection of entries describing an application

Device

Contact

Credential

Group

Application

Document

### Device Catalog

Special role is authorizing devices.

### Contact Catalog

Special role is access control on ingress

## Spools

### Outbound

Signed by the device

Resigned by the service dispatching

### Inbound

Service checks access control (veto)

Application checks access control (may be finer grain)

# Mesh Service

Untrusted service, minimize trust to bare minimum

Can't read content of any message.

Can only read contact catalog entries.

## Catalogs

### Account

Contains the account entries.

## Spools

### Account

Log of all updates to accounts.

Synchronization off-host provides backup.

### Incident

Reports of potential abuse

## Partitioning

Can split out handling of accounts to separate hosts each handling a subset of accounts.

User clients are directed to connect to a specific host in any case by means of redirects.

## Backup

Synchronizing Account spools protects data against loss and provides for fast restart capability.

# Security Considerations

The security considerations for use and implementation of Mesh services and applications are described in the Mesh Security Considerations guide <norm="draft-hallambaker-mesh-security"/>.

# IANA Considerations

This document does not contain actions for IANA

# Acknowledgements

Comodo Group: Egemen Tas, Melhi Abdulhayoğlu, Rob Stradling, Robin Alden.