Network Working Group
Internet-Draft
Intended status: Informational
Expires: October 13, 2018

P. Hallam-Baker
Comodo Group Inc.

April 11, 2018

Mathematical Mesh: Application Profiles
draft-hallambaker-mesh-app-02

Abstract

The use of the Mathematical Mesh to manage cryptographic keys for use with Mail and SSH is described. The format of the application profiles is described with examples.

This document is also available online at http://prismproof.org/Documents/draft-hallambaker-mesh-app.html [1] .

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 https://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 October 13, 2018.

Table of Contents

1. Introduction

2. Definitions

This section presents the related specifications and standard, the terms that are used as terms of art within the documents and the terms used as requirements language.

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 [RFC2119] .

2.2. Related Specifications

The related specifications are described in the Mesh Architecture specification [draft-hallambaker-mesh-architecture]

2.3. Defined Terms

No terms of art are defined.

2.4. Implementation Status

The implementation status of the reference code base is described in the companion document [draft-hallambaker-mesh-developer] .

3. Mesh Application Profiles

(Pull piece from Mesh Reference to here)

4. Catalog Profiles

Catalog profiles are used to synchronize encrypted data sets across devices. The catalog data model is restricted so as to permit a common set of management tools to be used to access and maintain profiles containing different types of data (bookmarks, credentials, contacts, etc.). Catalogs do not contain per device data. A catalog may not be shared with every device in the user?s profile but all the data in a catalog is available to all the devices with which it is shared.

The management operations supported are:

Synchronization Permit user to add, delete and update entries from multiple devices with minimal surprise. The mechanism is designed to be reasonably robust if network connectivity is lost during an attempted update.

Labelling Allow entries to be grouped into hierarchical categories defined by the user. An entry may be added to more than one category at once.

Each catalog entry SHOULD contain exactly one timestamp field of time Added, Updated or Deleted. If present, the timestamp entries and the entry identifiers are used to merge catalog profiles that have been updated separately leading to an inconsistent state.

Applications SHOULD specify a timestamp field on every entry unless it is known that update inconsistency cannot occur. For example, when initially populating a catalog.

4.1. Catalog Example

Alice creates a new bookmarks profile which is shared between her laptop and her phone. The initial profile is empty:

```
{
  "BookmarkProfilePrivate": {
    "Entries": []}}
```

Figure 1

Alice adds a bookmark entry to her profile on the browser on her laptop:

```
{
  "BookmarkProfilePrivate": {
    "Entries": [{
        "Added": "2018-04-11T17:01:08Z",
        "Title": "First Site",
        "Uri": "http://example.com/"}]}}
```

Figure 2

Later, Alice is attempting to connect to a site on her phone but has no network connection. She decides to bookmark the site instead.

```
{
  "BookmarkProfilePrivate": {
    "Entries": [{
        "Added": "2018-04-11T18:35:46Z",
        "Title": "Second Site",
        "Uri": "https://example.com/"}]}}
```

Figure 3

At this point, the profiles on Alice's two devices are out of sync. When the phone is finally able to connect to the network, the profiles are merged:

```
{
  "BookmarkProfilePrivate": {
    "Entries": [{
        "Added": "2018-04-11T17:01:08Z",
        "Title": "First Site",
        "Uri": "http://example.com/"},
      {
        "Added": "2018-04-11T18:35:46Z",
        "Title": "Second Site",
        "Uri": "https://example.com/"}]}}
```

Figure 4

4.2. Credentials

A credentials catalog contains access credentials, typically usernames and passwords, for a set of network resources such as Web sites that do not support the use of Mesh device profile data for authentication.

Mesh/Credential enabled applications SHOULD offer to generate strong passwords for the user if the AutoGenerate field is set to true in the credential profile. Since the use of automatically generated passwords is likely to be inconvenient for users unless all the applications on all the devices they might use support Mesh/Credential profiles, applications MUST NOT automatically generate passwords unless the user has affirmatively indicated that they want to use them.

Further Work: Credential entries MAY specify that the credential is restricted to use with certain protocols (Web browsing, SFTP, etc.) and/or certain authentication mechanisms but the precise means of identifying both is not currently defined.

4.2.1. Credentials Example

```
{
  "CredentialProfilePrivate": {
    "AutoGenerate": true,
    "Entries": [{
        "Sites": ["luggage.example.net"],
        "Username": "Alice",
        "Password": "12345"},
      {
        "Label": ["Linux"],
        "Sites": ["host.example.net"],
        "Username": "BitAlice",
        "Password": "password",
        "Protocol": "ssh"}],
    "NeverAsk": ["secure.example.com",
      "bank.example.com"]}}
```

Figure 5

4.3. Bookmarks

A bookmarks catalog contains a collection of bookmarks that have been saved for later use. While the ability share bookmarks between groups of users has obvious advantages, at present, the implementation and specification are only written with the use of a single user have been considered.

A bookmark entry contains the URI of the target and a title. If the book mark entry is a HTML resource, the title is taken from the <title> element in the document header. If network and storage resources permit, catalog entries MAY include a favicon value for easy identification.

Further Work: Bookmark entries MAY contain details describing the security properties of the connection to protect against downgrade attack. For example, information from HTTP strict security [RFC6797] and key pinning headers [RFC7460] .

tbs

4.4. Contacts

A contacts catalog contains a collection of contacts. The ContactEntry object contains the usual fields for describing the person or organization the entry refers to, and means of contact (Internet, Postal).

One significant deviation from existing formats is that the fact that people change names (e.g. marriage) is captured and that means of contact MAY be scoped to a particular organization.

4.4.1. Contacts Example

```
{
  "ContactProfilePrivate": {
    "Entries": [{
        "Personals": [{
            "First": "Alice"}],
        "Internets": [{
            "Uri": "mailto:alice@example.com"}]},
      {
        "Personals": [{
            "First": "Bob"}],
        "Internets": [{
            "Uri": "mailto:bob@example.com"}]}]}}
```

Figure 6

4.5. Calendar

It is generally acknowledged that representation of calendar information is a ?difficult? problem. Since it is the author?s experience that such problems almost invariably arise from an attempt to make use of an inadequate data model, the format for exchange of calendar information is currently undefined.

Further Work: Two major causes of difficulty are the use of local time zones and daylight savings, the definition of which are capricious at best. When a recurring meeting is specified it is vital that the time zone in which the meeting is to recur is specified explicitly. Attempts to normalize meetings to a single time zone will inevitably fail when the definition of time changes between the time the meeting is called and the meeting is held.

Another major limitation in existing formats is the lack of understanding that when the user travels, at least some part of their context for scheduling also changes. It should be possible to integrate all parts of the user?s schedule to offer alerts and reminders appropriate to their current location.

5. Mail

Mesh Mail profiles serve two distinct purposes:

- To provision a user?s devices with the credentials, network configuration and cryptographic keys necessary to support use of mail and end-to-end mail security enhancements.

- To publish necessary information for use by mail senders.

- While the principle focus of Mesh/Mail is to support exchange of mail over SMTP protocol, any infrastructure that provides a mechanism for publishing a recipient?s public keys for use by senders can, at least in principle, also publish information describing the user?s mail capabilities including the ability to support new messaging protocols.

- The use of end-to-end secure protocols requires the generation and use of at least one public key pair for signature and encryption. Best current practices require the use of separate keypairs for signature and encryption and if practical separate signature keys for each device.

- Since S/MIME and OpenPGP as currently specified do not support the use of Proxy Re-Encryption (recryption) to enable separate the use of separate decryption keys for each device, a single encryption keypair is used. A mail profile must therefore contain an encrypted copy of the corresponding decryption key for each device.

- Further Work: Support Signal etc. At present the profiles are not differentiated on a per device level. It is likely that it would be useful to specify that certain devices are to carry a complete copy of the user?s mail while others should only carry messages

from the last few weeks or months. It is also likely that it would be useful to be able to mark certain selections as being likely to be most useful offline.

5.1. Mail Example

6. SSH

The Secure Shell (SSH) transport layer protocol [RFC4253] is widely used as a mechanism for securing access to remote hosts. In addition to providing a terminal connection to a remote host, SSH also supports file transfer and remote access (VPN) functionality. It is also used to provide remote procedure call (RPC) capabilities in applications such as Git.

While SSH permits a high level of security to be achieved, achieving a high security configuration requires a considerable degree of attention to detail. Numerous ?how to? guides found on the Internet advise the user to engage in many unsafe practices. These include:

Using a single private key for authentication for every machine to be used as a client.

Emailing a copy of the authentication key to yourself to transfer it to a new machine. (Alternatively use of insecure FTP, copying the data to /temp, etc.)

Of equal concern was the fact that none of the guides mentioned any form of maintenance activity such as deleting authentication keys for a decommissioned device or performing a rekey operation in the case that a device is compromised.

Configuring SSH securely is a non-trivial task because SSH is the tool through which the administrator will be connecting to secure their system. This is a bootstrap problem: It is easy to solve the problem of SSH configuration once we have SSH configured for use. To enable SSH access to a machine without creating an insecure path first is not a trivial matter.

A Mesh/SSH profile contains three sets of information:

- A set of the user?s public authentication keys. This is used to generate auth_hosts files and equivalents to enable the user to access machines.

- A set of hosts known to the user. This is encrypted as it shows the machines that the user at least is likely to visit. This is

used to generate known_hosts files and equivalents to enable the user to authenticate hosts.

- o A set of device key entries. The entry for each host is encrypted. This is used to create the private key file(s) for the user on each of their devices.

6.1. SSH Example

7. Catalog Application Profiles

Catalogues are application profiles that consist of a set of related information (contacts, passwords, bookmarks) but do not contain any cryptographic private keys or device specific data. These restrictions allow management of these profiles to be simplified.

7.1. Shared

The following objects are common to multiple profiles.

7.1.1. Structure: ApplicationProfileCatalog

Inherits: ApplicationProfile

Base class for all application profiles that are tied to an account profile

AccountIdentifier: String (Optional) The account to which this profile is bound

PersonalUDF: String (Optional) The person to which this profile is bound

7.1.2. Structure: CatalogEntry

Base class for catalog entries, contains base information on which catalog operations are performed.

ID: String (Optional) Unique identifier for the entry. If present, overrides the identifier specified in the entry.

Added: DateTime (Optional) The time the site was added

Updated: DateTime (Optional) The last time the entry was updated

Deleted: DateTime (Optional) The last time the entry was updated

Label: String [0..Many] Labels identifying the group(s) that the entry is filed under

Source: TypedData [0..Many] Source data for the entry

7.1.3. Structure: TypedData

Typed content.

ContentType: String (Optional) IANA Content Type identifier

Data: Binary (Optional) The described data

7.2. Credential Catalog

Profile for recording access credentials for Web sites and other projects. Currently this is limited to usernames and passwords but could expand to include other credential forms.

7.2.1. Structure: CredentialProfile

Inherits: ApplicationProfileCatalog

Stores usernames and passwords. There are no public fields.

[No fields]

7.2.2. Structure: CredentialProfilePrivate

Inherits: ApplicationProfilePrivate

Private part of the profile.

AutoGenerate: Boolean (Optional) If true, a client MAY offer to automatically generate strong (i.e. not memorable) passwords for a user. A user would not normally want to use this feature unless they have access to Mesh password management on every device they use to browse the Web

Entries: CredentialEntry [0..Many] A list of password credential entries.

NeverAsk: String [0..Many] A list of domain names of sites for which clients MUST NOT ask to store passwords for.

7.2.3. Structure: CredentialEntry

Inherits: CatalogEntry

Username password entry for a single site

Sites: String [0..Many] DNS name of site *.example.com matches www.example.com etc.

Username: String (Optional) Case sensitive username

Password: String (Optional) Case sensitive password.

Protocol: String (Optional) Protocol identifier, e.g. http, sftp, ssh, etc.

7.3. Bookmark Catalog

Profile for recording Web site bookmarks and related information.

7.3.1. Structure: BookmarkProfile

Inherits: ApplicationProfileCatalog

Stores Web site bookmarks in a hierarchical

[No fields]

7.3.2. Structure: BookmarkProfilePrivate

Inherits: ApplicationProfilePrivate

Private part of the profile.

Entries: BookmarkEntry [0..Many] The bookmark entries

7.3.3. Structure: BookmarkEntry

Inherits: CatalogEntry

Bookmark entry for a single site

Title: String (Optional) The resource name

Uri: String (Optional) The resource identifier

ImageUDF: String [0..Many] UDF fingerprint of related favicon image

7.4. Contact Catalog

Profile for recording user contact information

7.4.1. Structure: ContactProfile

Inherits: ApplicationProfileCatalog

Stores Web site bookmarks in a hierarchical

[No fields]

7.4.2. Structure: ContactProfilePrivate

Inherits: ApplicationProfilePrivate

Private part of the profile.

Entries: ContactEntry [0..Many] The contact entries

7.4.3. Structure: ContactEntry

Inherits: CatalogEntry

Contact entry

Personals: PersonalName [0..Many] Personal names.

MeshUDFs: String [0..Many] List of mesh profiles fingerprints for the user.

Internets: Internet [0..Many] List of Internet, telephone, etc addresses for contacting this party

Postals: Postal [0..Many] List of postal addresses for this party

7.4.4. Structure: PersonalName

Personal name structure.

First: String (Optional) First name

Last: String (Optional) Last name

Midle: String (Optional) Middle names (if used).

7.4.5. Structure: Address

Contact address.

Label: String [0..Many] Labels identifying the modes in which the label may be used e.g. Home, Business, Mobile

Attributes: String [0..Many] Attributes describing the mode in which the contact address may be used.

7.4.6. Structure: Internet

Internet contact address

Inherits: Address

Inherits: Address

Uri: String (Optional) The resource identifier describing the mode of contact

7.4.7. Structure: Postal

Postal or geographic address.

Inherits: Address

Inherits: Address

Adressee: String (Optional) The postal name

Street: String (Optional) Street name and number

Town: String (Optional) Name of town or city

Region: String (Optional) State, county, department or other government unit.

Country: String (Optional) The country name

Code: String (Optional) The ISO 3 letter country code

7.4.8. Structure: ContactPerson

Inherits: ContactEntry

Contact entry for a single person

FullName: String (Optional) The name of the person

Organization: String [0..Many] The name of the organizations the person is associated with

7.4.9. Structure: ContactOrganization

Inherits: ContactEntry

Contact entry for a single organization

FullName: String (Optional) The name of the organization

7.4.10. Structure: NetworkProfile

Inherits: ApplicationProfileCatalog

Stores usernames and passwords. There are no public fields.

[No fields]

7.4.11. Structure: NetworkProfilePrivate

Inherits: ApplicationProfilePrivate

Private part of the profile.

AccessPoints: NetworkEntry [0..Many] A list of access point entries

VPNs: NetworkEntry [0..Many] A list of VPN entries

7.4.12. Structure: NetworkEntry

Inherits: CredentialEntry

Describes network access credentials

Configuration: String (Optional) Network configuration data.

7.5. Mail Application Profile Objects

Profiles that describe mail user agent configuration

7.5.1. Structure: MailProfile

Inherits: ApplicationProfile

Public profile describes mail receipt policy. Private describes Sending policy

EncryptionPGP: PublicKey (Optional) The current OpenPGP encryption key

EncryptionSMIME: PublicKey (Optional) The current S/MIME encryption key

7.5.2. Structure: MailDevicePublic

Contains public device description

Inherits: ApplicationDevicePublic

[No fields]

7.5.3. Structure: MailProfilePrivate

Inherits: ApplicationProfilePrivate

Describes a mail account configuration

Private profile contains connection settings for the inbound and outbound mail server(s) and cryptographic private keys. Public profile may contain security policy information for the sender.

EmailAddress: String (Optional) The RFC822 Email address. [e.g. "alice@example.com"]

ReplyToAddress: String (Optional) The RFC822 Reply toEmail address. [e.g. "alice@example.com"]

When set, allows a sender to tell the receiver that replies to this account should be directed to this address.

DisplayName: String (Optional) The Display Name. [e.g. "Alice Example"]

AccountName: String (Optional) The Account Name for display to the app user [e.g. "Work Account"]

Inbound: Connection [0..Many] The Inbound Mail Connection(s). This is typically IMAP4 or POP3

If multiple connections are specified, the order in the sequence indicates the preference order.

Outbound: Connection [0..Many] The Outbound Mail Connection(s). This is typically SMTP/SUBMIT

If multiple connections are specified, the order in the sequence indicates the preference order.

Sign: PublicKey [0..Many] The public keypair(s) for signing and decrypting email.

If multiple public keys are specified, the order indicates preference.

Encrypt: PublicKey [0..Many] The public keypairs for encrypting and decrypting email.

If multiple public keys are specified, the order indicates preference.

7.5.4. Structure: MailDevicePrivate

Private data specific to the device

Inherits: ApplicationDevicePrivate

[No fields]

7.6. SSH Application Profile Objects

Profiles that describe SSH user agent configuration

7.6.1. Structure: SSHProfile

Application profile for SSH. This is an initial cut of the profile and will need revision. In particular, a sysadmin with a very large number of hosts they are accessing will need some means of avoiding combinatorial explosion.

Inherits: ApplicationProfile

[No fields]

7.6.2. Structure: SSHDevicePublic

Contains public device description

Inherits: ApplicationDevicePublic

Inherits: ApplicationDevicePublic

PublicKey: PublicKey (Optional) Public authentication key for a device.

7.6.3. Structure: SSHProfilePrivate

Private portion or profile.

Inherits: ApplicationProfilePrivate

Inherits: ApplicationProfilePrivate

Account: String (Optional) The account to which the profile is bound

HostEntries: HostEntry [0..Many] Hosts bound to the profile

7.6.4. Structure: HostEntry

Describe a host connected to the SSH profile. This is a machine that the user will access using the credential.

Inherits: Entry

Inherits: Entry

Address: String (Optional) The DNS address or IP address of the host

AlgorithmID: String (Optional) The SSH Algorithm identifier

PublicKey: String (Optional) The Base64 encoded public key

7.6.5. Structure: SSHDevicePrivate

Private data specific to the device

Inherits: ApplicationDevicePrivate

Inherits: ApplicationDevicePrivate

DevicePrivateKey: PublicKey (Optional) A private keypair or keypair contribution created for exclusive use of this device.

KeyUDF: String (Optional) Fingerprint of device that this key corresponds to.

8. Acknowledgements

Your name could appear here.

9. Security Considerations

[This is just a sketch for the present.]

10. IANA Considerations

[TBS list out all the code points that require an IANA registration]

11. References

11.1. Normative References

[draft-hallambaker-mesh-architecture]
Hallam-Baker, P., "Mathematical Mesh: Architecture", draft-hallambaker-mesh-architecture-04 (work in progress), September 2017.

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

[RFC4253] Ylonen, T. and C. Lonvick, "The Secure Shell (SSH) Transport Layer Protocol", RFC 4253, DOI 10.17487/RFC4253, January 2006.

[RFC6797] Hodges, J., Jackson, C., and A. Barth, "HTTP Strict Transport Security (HSTS)", RFC 6797, DOI 10.17487/RFC6797, November 2012.

[RFC7460] Chandramouli, M., Claise, B., Schoening, B., Quittek, J., and T. Dietz, "Monitoring and Control MIB for Power and Energy", RFC 7460, DOI 10.17487/RFC7460, March 2015.

11.2. Informative References

[draft-hallambaker-mesh-developer]
Hallam-Baker, P., "Mathematical Mesh: Reference Implementation", draft-hallambaker-mesh-developer-06 (work in progress), April 2018.

11.3. URIs

[1] http://prismproof.org/Documents/draft-hallambaker-mesh-app.html

Author's Address

Phillip Hallam-Baker
Comodo Group Inc.

Email: philliph@comodo.com