Mathematical Mesh: Application Profiles

Mesh/SSH

<series>draft-hallambaker-mesh-app-ssh

<status>informational

<stream>independent

<ipr>trust200902

<author>Phillip Hallam-Baker

<surname>Hallam-Baker

<initials>P. M.

<firstname>Phillip

<organization>Comodo Group Inc.

<email>[philliph@comodo.com](mailto:philliph@comodo.com)

<also>http://prismproof.org/Documents/draft-hallambaker-mesh-app-ssh.html

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.

# Introduction

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

The Mathematical Mesh provides an infrastructure for single touch administration of all the devices a user has connected to their Mesh profile. Managing SSH configuration with the Mesh guides the user towards use of a maximally secure configuration. Once the configuration is achieved, it is maintained automatically.

# 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.

## 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 <norm="RFC2119"/>.

## Related Specifications

The related specifications are described in the Mesh Architecture specification <norm="draft-hallambaker-mesh-architecture"/>

## Defined Terms

No terms of art are defined.

## Implementation Status

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

# User Experience

Alice is a user of three machines, X, Y and Z. Machines X and Y are configured with monitors and keyboards enabling their use as an SSH client. Machine Z is to be configured as a headless machine (no monitor). When finished, Alice wants to be able to connect to machines X, Y or Z from machine X or Y.

Since SSH is a tool principally used on UNIX based systems and machines configured to provide similar behavior, this guide assumes machines X, Y and Z are all UNIX systems when describing the files to be modified. When used on a Windows machine, the Mesh tools configure the equivalent Windows files.

## Initial profile creation

If this is the first time Alice has used the Mesh, she creates a personal profile for herself on machine X:

X> meshman /personal alice@mesh.prismproof.org

## Configuring SSH for machine X

To configure SSH on machine X, Alice adds it to her profile.

X> meshman /ssh

Note that the meshman tool only performs the

At this point, Alice has a new private key that is unique to machine X and the corresponding public key has been added to her profile

## Configuring SSH for machine Y

To configure the second machine, Alice first requests adding it to her profile:

Y>

This request must be accepted on machine X:

X>

Alice now adds the SSH profile to machine Y:

Y> meshman /ssh

At this point machines X and Y both have a unique private key and both the corresponding public keys have been added to the authorized key files on Y. The authorized key file on X will be updated by a mesh profile manager running periodically.

## Configuring SSH for machine Z

Configuration of machine Z begins in exactly the same way as for machine Y. The only difference is that when she creates the SSH profile for the device, she requests it be a host only profile:

Z> meshman /ssh /host

## Deleting a Device

## Future directions

### Native Mesh support in SSH

The chief weakness in this user experience is that the machines cannot update themselves automatically or even know when an update is required. The only approach that is available is for the host devices to periodically poll the Mesh portal and request updates for the registered profiles.

This delay can be avoided if the SSH protocol and implementations were updated to support direct use of Mesh profiles. This allows a client to push the updated profile data to the server when making the connection attempt.

### Update Daemon

Native support allows the process of adding devices to be automated but does not guarantee timely processing of deletion requests. It also requires action by third parties who may not be interested in providing Mesh support.

A better approach would be to run a daemon on each machine that could receive update notifications from the portal whenever a significant event (profile addition/deletion) had occurred.

### Batched Connection Requests

The existing management tools require a device to be connected to a profile before use of applications are enabled on the device.

### Enhanced administration

While this approach is functional, it does not meet the requirement for complete mediation of the log in request. While the system administrator has the ability to permit or deny remote access capabilities to a user, these are the only options that the administration tools currently supported by most SSH configurations provide. The administrator cannot control the types of credentials used by specific users.

The Mesh could be employed to permit a closer level of control. Storing the user’s profile fingerprint in the password database (/etc/passwd or /etc/shadow) would allow every application and system utility to refer to a single root of trust for authenticating every credential presented by a user.

# Platform Bindings

Since SSH is an application program

OpenSSH Files

authorized\_keys.pub

Contains a list of all the keys that are authorized to access the corresponding account. This is a list of Mesh SSH per device application keys.

Id\_rsa.pub

Default name for the user’s RSA public key

Id\_rsa

Default name for the user’s RSA private key

known\_hosts

List of host keys known to this user.

Application Profile

# Acknowledgements

# Security Considerations

[This is just a sketch for the present.]

# IANA Considerations

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