|  |  |
| --- | --- |
| Current version | 1.0 DRAFT |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description of change |
| 1.0 | 6 October 2015 | Sriram Nambakam | Initial draft |

**Reviewers and Signoff History**

|  |  |  |  |
| --- | --- | --- | --- |
| Reviewer | Role | Comments | Date |
|  |  |  |  |

2. Summary 3

2.1. Overview 3

2.2. Purpose 3

2.2.1. Business Driver(s) 3

2.2.2. Technical Driver(s) 3

2.3. Goal 3

3. Problem Description 3

3.1. Problem 3

3.2. Current Product Limitations 3

3.3. Use Cases 3

4. Product Requirements 3

4.1. Compatibility 3

4.2. Software Lifecycle 3

4.3. Installation 3

4.4. Deployment 3

4.5. Third party component deployment 3

4.6. Upgrade 3

4.7. Legacy 3

4.8. Future 3

4.9. Backup/Restore 3

4.10. Performance and Scalability 3

4.11. Large Scale Performance Considerations 3

4.12. Performance and Scalability Targets 3

4.13. Feature Interoperability **Error! Bookmark not defined.**

5. Other Considerations 3

5.1. Licensing 3

5.2. Disk footprint 3

5.3. Memory footprint 3

# Summary

## Overview

This document provides the specification for a minimal Representational State Transfer (ReST) engine that can be embedded in services that are implemented in ANSI C.

## Purpose

### Business Driver(s)

User interfaces and other services require a HTTP(S) based REST interface for easy integration with our various existing services that are implemented in ANSI C.

A ReST interface is preferred to easily negotiate proxies, firewalls and also provide easier integration with cloud enabled services.

### Technical Driver(s)

The primary technical drivers are the following.

* Size
* Performance
* Extensibility

## Goal

The primary deliverable is a shared library for the ReST engine. It will be supported on the following platforms.

* Linux
* Windows
* Mac

The ReST Engine will support the following protocols.

* HTTP
* HTTPS

The following payload format will be supported.

* Javascript Object Notation (JSON)

The following components will be provided to support the ReST engine.

* A multi-threaded server that consumes the services of the ReST engine
* A multi-threaded client that communicates with the server
* The server and client will support a ReST API that utilizes a message format implemented using JSON.

# Problem Description

## Problem

## Current Product Limitations

The ReST engine must be implemented using ANSI C.

All encryption must be handled through OpenSSL.

## Use Cases

### LDAP over REST for VMware Directory

The VMware Directory Service will serve LDAP requests over ReST. In order to achieve this, the directory service will embed the ReST engine.

### REST interface for VMware Authentication Framework

TBD

### REST interface for VMware Certificate Authority

TBD

# Product Requirements

## Compatibility

## Software Lifecycle

### Installation

### Deployment

#### Third party component deployment

### Upgrade

#### Legacy

#### Future

### Backup/Restore

## Performance and Scalability

### Large Scale Performance Considerations

### Performance and Scalability Targets

## User interfaces

# Other Considerations

## Licensing

The code will be distributed using the Apache 2.0 license

## Disk footprint

TBD

## Memory footprint

TBD

# References