Hot Protocol Version 2 (HotPv2)

Hottentot RPC Framework

Kamran Amini

June 4, 2016

Contents

1 Introduction

This document talks about request and response structures and mechanisms in Hottentot RPC Framework. Purpose of this protocol is to convey Method Invocation request and response. Current protocol is serialization transparent and can convey a method call with arguments produced with different serialization algorithms. In this version, Hottentot's runtimes can only work with internal serialization mechanism.

2 Glossary

SERIALIZATION
A process in which an object turns into a byte array to be transferred using a channel.

STRUCT
A term used for encapsulation of fields related to a specific entity. It is a structure and it will be generated differently for each programming language.

IDL (INTERFACE DEFINITION LANGUAGE)
An IDL is a language transparent to all programming languages which Hottentot supports. IDL can be generated to any target languages supported by Hottentot RPC Framework.

HOT FILE
A file which contains IDL. Hot files usually end with `.hot` extension.

GENERATOR
A tool for generating stub and struct source codes for a target programming language. Currently, generators for C++ and Java languages are available.

RUNTIME
A library for a specific programming language which performs Service and Proxy operations. Currently, runtimes are only available for C++ and Java.

ENDPOINT
Endpoint is a combination of IP address and a port. One service object or many can be bound to an endpoint.

CURRENT ENDPOINT
When talking about a proxy, Current Endpoint is an endpoint which client has used for connecting to server-side.

SERVICE
Service is an object serving method invocation requests.

PROXY
Proxy is an object which produces method invocation and other types of requests and receives the response. A proxy object talks to an endpoint at first and its request will be delegated to a service object in case of method invocation requests.

PROXY-SIDE
A software or library which tries to interact with service-side objects like endpoints, services, etc.

SERVICE-SIDE
A software or library which serves proxy-side requests and generates suitable response.

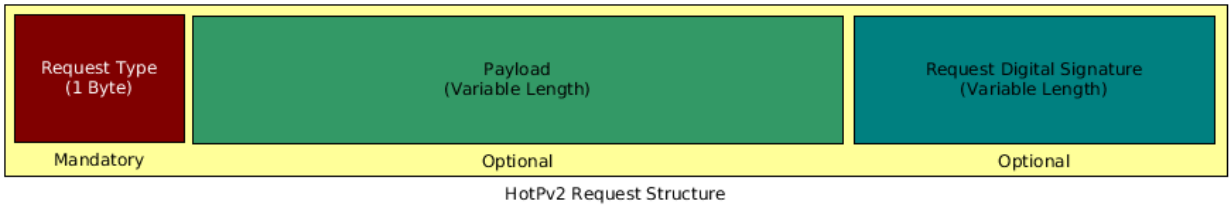
3 Request

Each request consists of 1 mandatory and 2 optional fields.

- **Request Type**: This field is mandatory. Using this field determines type of request.
- **Payload**: This files is optional and carries all necessary data to perform the request operation.
- **Request Digital Signature** This fields is optional and carries a PKI compatible digital signature over Request Type and Payload.

Below figure shows the request structure.

Figure 1: HotPv2 Request Structure



3.1 Request Type

Determines the request type and payload structure. Following C enumeration shows the possible values for this field.

```
enum RequestType {
  Unknown = 0,
  ServiceListQuery = 1,
  MethodInvocation = 2,
  MethodListQuery = 3,
  ServiceInfoQuery = 4,
  EndpointInfoQuery = 5,
  ProxyInfo = 6
};
```

Values can be:

- **Unknown**: It means nothing to Hottentot service side and these requests should be ignored by the implementation.
- **ServiceListQuery**: Proxy-side queries about the list of available services. Hottentot service runtime should return list of services exposed on current endpoint.
- **MethodInvocation**: Invokes a method on a specific service object.
- **MethodListQuery**: Proxy-side queries about the list of callable methods on a specific service object.
- **ServiceInfoQuery**: Proxy-side asks about parameters of a specific service.
- **EndpointInfoQuery**: Proxy-side asks about parameters of current endpoint.
- **ProxyInfo**: Proxy-side sends its parameters to current endpoint.

3.2 Payload

Requests can have payloads. Many request types need data for their operations and payload part carries the data. Following sections explain about payload structure for each request type.

3.2.1 Payload for MethodInvocation Request Type

A Method Invocation request payload consists of following fields:

- Service Id (4 Bytes)
- Method Id (4 Bytes)
- Number of Arguments (1 Byte)
- Arguments as an array of LV Structures. (Variable Length)

Figure 2: HotPv2 Method Invocation Request Structure

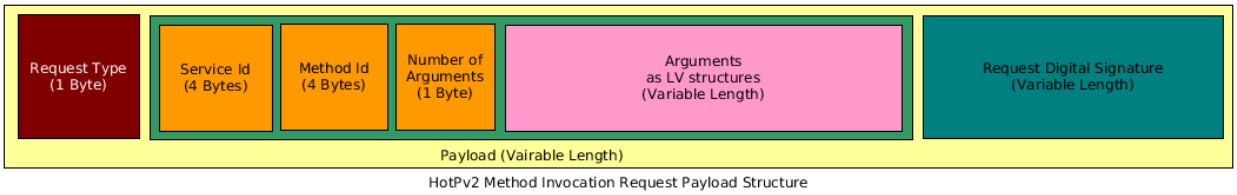
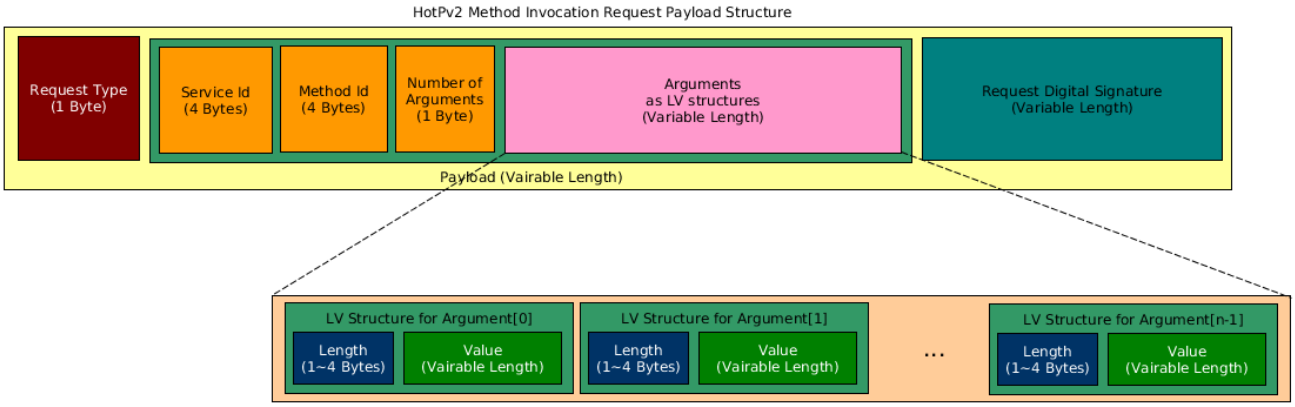


Figure 2 shows the structure of a request. Arguments can be transferred using LV structures. LV structures include Length and Value. Maximum length for a single LV structure is $2^{32} - 1$ since length values can be stored in a field at most 4 bytes. Figure 3 shows the LV structures in detail.

Figure 3: Structure of a Method Invocation request payload with arguments' LV structures.



Value can be any byte array but usually it is a serialized object. Serialization method can be anything. Hottentot itself provides an algorithm for serialization and `serialize()` and `deserialize()` methods are generated for every struct. Current stub generation mechanism works only with Hottentot's internal serialization.

4 Future Features

- Version should be added to request and response structures.