Benchmarking gRPC for Redfish

Bailey Hollis – Texas A&M University

Jeff Hilland – HPE Labs

OCP GLOBAL SUMMIT

October 13-16, 2025
San Jose, California

Introduction

Redfish

- API used to manage servers and equipment
- Wide industry backing, evolving scope

REST

- Approachable for developers, supports HTTP/1.1 and HTTP/2
- Payload is human readable, machine capable

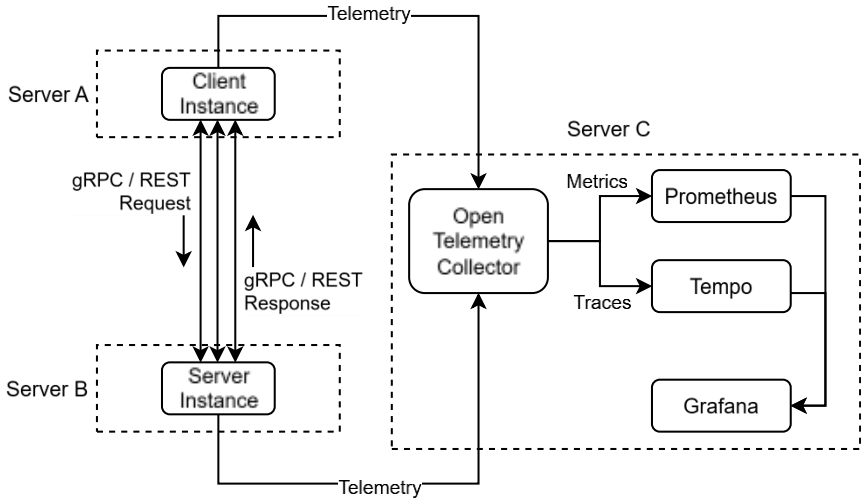
gRPC

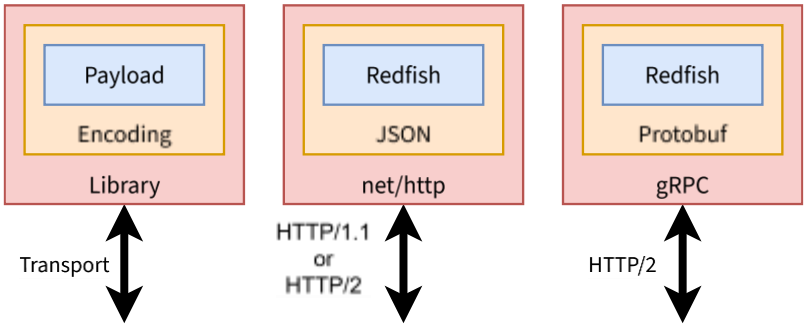
- Possible new approach for Redfish transport
- Claimed efficiency gains and improved throughput
- Payload not human readable (Protocol Buffers / Protobuf)

Provide data to answer the question:
Is gRPC a suitable transport for the Redfish data model?

Testing Methodology

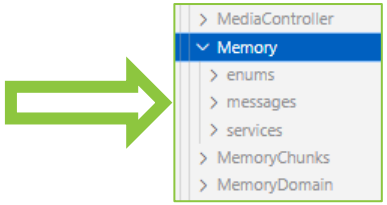
1. REST Server + Client implementation
2. Develop Redfish Protobuf encoding
3. Extend implementation with gRPC
4. Conduct comprehensive benchmarks

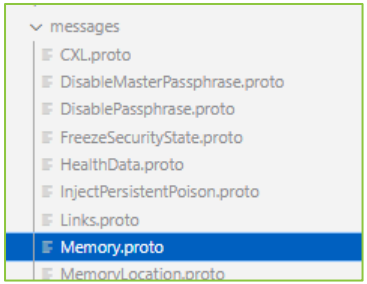




Creating the Protobuf Encoding

Redfish Schema





Protobuf Definition

Generating a Redfish Library

```
message Memory {
    optional Redfish.odata.context odatacontext = 1 [json_name = "@odata.context"]; //  ['odata', 'context']
    optional Redfish.odata.etag odataetag = 2 [json_name = "@odata.etag"]; //  ['odata', 'etag']
    optional Redfish.odata.id odataid = 3 [json_name = "@odata.id"]; //  ['odata', 'id']
    optional Redfish.odata.type odatatype = 4 [json_name = "@odata.type"]; //  ['odata', 'type']
    optional int32 allocation_alignment_mi_b = 5 [json_name = "AllocationAlignmentMiB"]; // The boundary that mem
    optional int32 allocation_increment_mi_b = 6 [json_name = "AllocationIncrementMiB"]; // The size of the small
    repeated int32 allowed_speeds_m_hz = 7 [json_name = "AllowedSpeedsMHz"]; // Speeds supported by this memory d
    optional Redfish.odata.id assembly = 8 [json_name = "Assembly"]; // The link to the assembly resource associa
    optional BaseModuleType base_module_type = 9 [json_name = "BaseModuleType"]; // The base module type of the m
    optional int32 bus_width_bits = 10 [json_name = "BusWidthBits"]; // The bus width, in bits.
    optional CXL c_x_l = 11 [json_name = "CXL"]; // CXL properties for this memory device. ['Memory', 'CXL']
    optional int32 cache_level = 12 [json_name = "CacheLevel"]; // The level of the cache memory.
    optional int32 cache_size_mi_b = 13 [json_name = "CacheSizeMiB"]; // Total size of the cache portion memory i
    optional int32 capacity_mi_b = 14 [json_name = "CapacityMiB"]; // Memory capacity in mebibytes (MiB).
    optional Redfish.odata.id certificates = 15 [json_name =
    optional bool configuration_locked = 16 [json_name = "Co
    optional int32 data_width_bits = 17 [json_name = "DataWi
```

Protobuf Definition

Generated Go Code

```
// Message Definition
type Memory struct {
    state                protoimpl.MessageState          `protogen:"open.v1"`
    Odatacontext         *odata.Context                  `protobuf:"bytes,1,opt,name=odatacontext
    Odataetag            *odata.Etag                     `protobuf:"bytes,2,opt,name=odataetag,js
    Odataid              *odata.Id                       `protobuf:"bytes,3,opt,name=odataid,json
    Odatatype            *odata.Type                     `protobuf:"bytes,4,opt,name=odatatype,js
    AllocationAlignmentMiB *int32                        `protobuf:"varint,5,opt,name=allocation_
    AllocationIncrementMiB *int32                        `protobuf:"varint,6,opt,name=allocation_
    AllowedSpeedsMHz     []int32                         `protobuf:"varint,7,rep,packed,name=allo
    Assembly             *odata.Id                       `protobuf:"bytes,8,opt,name=assembly,jso
    BaseModuleType       *BaseModuleType.BaseModuleType  `protobuf:"varint,9,opt,name=base_module
    BusWidthBits         *int32                          `protobuf:"varint,10,opt,name=bus_width_
    CXL                  *CXL.CXL                        `protobuf:"bytes,11,opt,name=c_x_l,json=
    CacheLevel           *int32                          `protobuf:"varint,12,opt,name=cache_leve
    CacheSizeMiB         *int32                          `protobuf:"varint,13,opt,name=cache_size
    CapacityMiB          *int32                          `protobuf:"varint,14,opt,name=capacity_m
    Certificates         *odata.Id                       `protobuf:"bytes,15,opt,name=certificate
    ConfigurationLocked  *bool                           `protobuf:"varint,16,opt,name=configurat
    DataWidthBits        *int32                          `protobuf:"varint,17,opt,name=data_width
```

Storing a Mockup Object

```
{
    "@odata.type": "#Memory.v1_21_0.Memory",
    "Id": "DIMM1",
    "Name": "DIMM Slot 1",
    "RankCount": 2,
    "MaxTDPMilliWatts": [
        12000
    ],
    "CapacityMiB": 32768,
    "DataWidthBits": 64,
    "BusWidthBits": 72,
    "ErrorCorrection": "MultiBitECC",
    "MemoryLocation": {
        "Socket": 1,
        "MemoryController": 1,
        "Channel": 1,
        "Slot": 1
    },
    "Location": {
        "PartLocation": {
            "ServiceLabel": "DIMM 1",
            "LocationType": "Slot",
            "LocationOrdinalValue": 0
        }
    },
    "MemoryType": "DRAM",
    "MemoryDeviceType": "DDR4",
    "BaseModuleType": "RDIMM",
    "MemoryMedia": [
        "DRAM"
    ],
    "Status": {
        "State": "Enabled",
        "Health": "OK"
    },
    "EnvironmentMetrics": {
        "@odata.id": "/redfish/v1/.../EnvironmentMetrics"
    },
    "@odata.id": "/redfish/v1/Systems/437XR1138R2/Memory/DIMM1",
    "@Redfish.Copyright": "Copyright 2014-2025 DMTF."
}
```

Redfish Mockup

```
var demoMemoryProto Memory.Memory = Memory.Memory{
    Odatatype:          &odata.Type{Type: "#Memory.v1_21_0.Memory"},
    Id:                 &Resource.Id{Id: proto.String("DIMM1")},
    Name:               &Resource.Name{Name: proto.String("DIMM Slot 1")},
    RankCount:          proto.Int32(2),
    MaxTDPMilliWatts: []int32{12000},
    CapacityMiB:        proto.Int32(32768),
    DataWidthBits:      proto.Int32(64),
    BusWidthBits:       proto.Int32(72),
    ErrorCorrection:  ErrorCorrection.ErrorCorrection_ERROR_CORRECTION_MULTI_BIT_E_C_C.Enum(),
    MemoryLocation: &MemoryLocation.MemoryLocation{
        Socket:           proto.Int32(1),
        MemoryController: proto.Int32(1),
        Channel:          proto.Int32(1),
        Slot:             proto.Int32(1),
    },
    Location: &Resource.Location{
        PartLocation: &Resource.PartLocation{
            ServiceLabel:         proto.String("DIMM 1"),
            LocationType:         Resource.LocationType_LOCATION_TYPE_SLOT.Enum(),
            LocationOrdinalValue: proto.Int32(0),
        },
    },
    MemoryType:       MemoryType.MemoryType_MEMORY_TYPE_D_R_A_M.Enum(),
    MemoryDeviceType: MemoryDeviceType.MemoryDeviceType_MEMORY_DEVICE_TYPE_D_D_R4.Enum(),
    BaseModuleType:   BaseModuleType.BaseModuleType_BASE_MODULE_TYPE_R_D_I_M_M.Enum(),
    MemoryMedia: []MemoryMedia.MemoryMedia{
        MemoryMedia.MemoryMedia_MEMORY_MEDIA_D_R_A_M,
    },
    Status: &Resource.Status{
        State:  Resource.State_STATE_ENABLED.Enum(),
        Health: Resource.Health_HEALTH_OK.Enum(),
    },
    EnvironmentMetrics: &odata.Id{Id: "/redfish/v1/.../EnvironmentMetrics"},
    Odataid:            &odata.Id{Id: "/redfish/v1/Systems/437XR1138R2/Memory/DIMM1"},
}
```

Protobuf Message Object

Mockup Object Wire Format

```
var demoMemoryProto Memory.Memory = Memory.Memory{
    Odatatype:        &odata.Type{Type: "#Memory.v1_21_0.Memory"},
    Id:               &Resource.Id{Id: proto.String("DIMM1")},
    Name:             &Resource.Name{Name: proto.String("DIMM Slot 1")},
    RankCount:        proto.Int32(2),
    MaxTDPMilliWatts: []int32{12000},
    CapacityMiB:      proto.Int32(32768),
    DataWidthBits:    proto.Int32(64),
    BusWidthBits:     proto.Int32(72),
    ErrorCorrection:  ErrorCorrection.ErrorCorrection_ERROR_CORRECTION_MULTI_BIT_E_C_C.Enum(),
    MemoryLocation: &MemoryLocation.MemoryLocation{
        Socket:           proto.Int32(1),
        MemoryController: proto.Int32(1),
        Channel:          proto.Int32(1),
        Slot:             proto.Int32(1),
    },
    Location: &Resource.Location{
        PartLocation: &Resource.PartLocation{
            ServiceLabel:        proto.String("DIMM 1"),
            LocationType:        Resource.LocationType_LOCATION_TYPE_SLOT.Enum(),
            LocationOrdinalValue: proto.Int32(0),
        },
    },
    MemoryType:       MemoryType.MemoryType_MEMORY_TYPE_D_R_A_M.Enum(),
    MemoryDeviceType: MemoryDeviceType.MemoryDeviceType_MEMORY_DEVICE_TYPE_D_D_R4.Enum(),
    BaseModuleType:   BaseModuleType.BaseModuleType_BASE_MODULE_TYPE_R_D_I_M_M.Enum(),
    MemoryMedia: []MemoryMedia.MemoryMedia{
        MemoryMedia.MemoryMedia_MEMORY_MEDIA_D_R_A_M,
    },
    Status: &Resource.Status{
        State:  Resource.State_STATE_ENABLED.Enum(),
        Health: Resource.Health_HEALTH_OK.Enum(),
    },
    EnvironmentMetrics: &odata.Id{Id: "/redfish/v1/.../EnvironmentMetrics"},
    Odataid:            &odata.Id{Id: "/redfish/v1/Systems/437XR1138R2/Memory/DIMM1"},
}
```

Protobuf Message Object

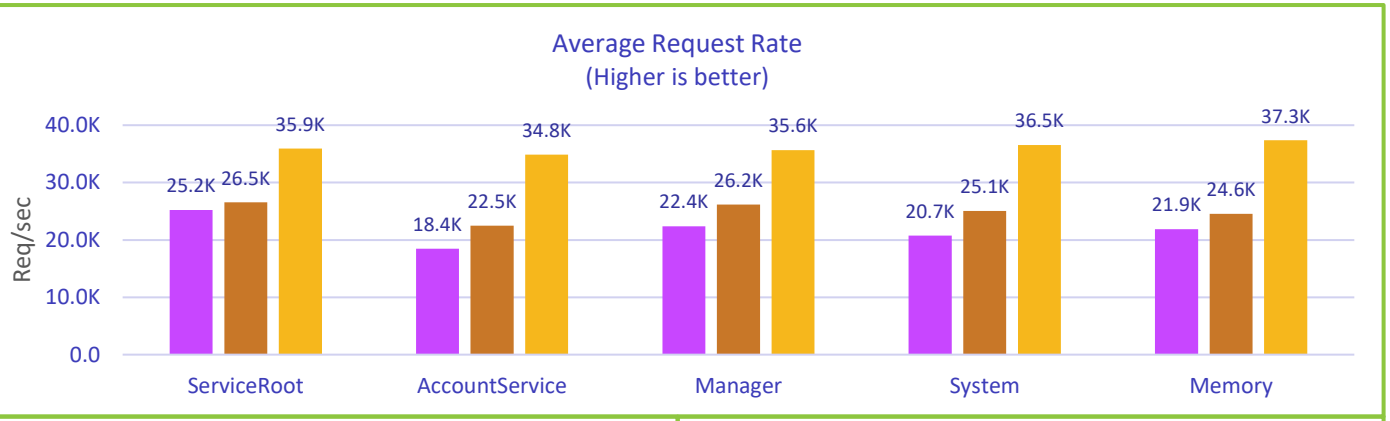
```
{
    "@odata.id": { "id": "/redfish/v1/Systems/437XR1138R2/Memory/DIMM1" },
    "@odata.type": { "type": "#Memory.v1_21_0.Memory" },
    "BaseModuleType": "RDIMM",
    "BusWidthBits": 72,
    "CapacityMiB": 32768,
    "DataWidthBits": 64,
    "EnvironmentMetrics": { "id": "/redfish/v1/.../EnvironmentMetrics" },
    "ErrorCorrection": "MultiBitECC",
    "Id": { "Id": "DIMM1" },
    "Location": {
        "PartLocation": {
            "LocationOrdinalValue": 0,
            "LocationType": "Slot",
            "ServiceLabel": "DIMM 1"
        }
    },
    "MaxTDPMilliWatts": [ 12000 ],
    "MemoryDeviceType": "DDR4",
    "MemoryLocation": {
        "Channel": 1,
        "MemoryController": 1,
        "Slot": 1,
        "Socket": 1
    },
    "MemoryMedia": [ "DRAM" ],
    "MemoryType": "DRAM",
    "Name": { "Name": "DIMM Slot 1" },
    "RankCount": 2,
    "Status": {
```

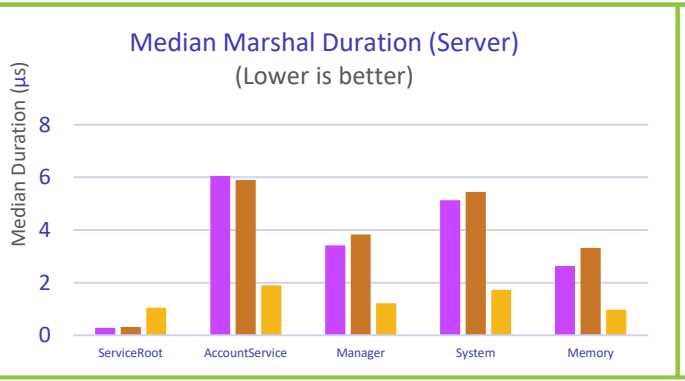
Json Encoding

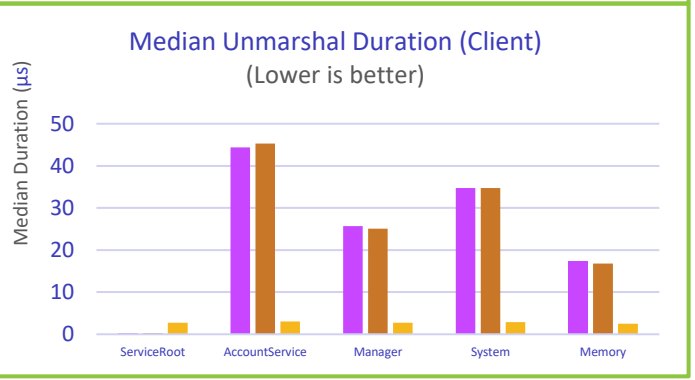
```
0000  00 00 00 01 4a 0a 99 02  0a 29 74 79 70 65 2e 67   ····J···· ·)type.g
0010  6f 6f 67 6c 65 61 70 69  73 2e 63 6f 6d 2f 52 65   oogleapi s.com/Re
0020  64 66 69 73 68 2e 4d 65  6d 6f 72 79 2e 4d 65 6d   dfish.Me mory.Mem
0030  6f 72 79 12 eb 01 1a 2e  0a 2c 2f 72 65 64 66 69   ory····· ·,/redfi
0040  73 68 2f 76 31 2f 53 79  73 74 65 6d 73 2f 34 33   sh/v1/Sy stems/43
0050  37 58 52 31 31 33 38 52  32 2f 4d 65 6d 6f 72 79   7XR1138R 2/Memory
0060  2f 44 49 4d 4d 31 22 18  0a 16 23 4d 65 6d 6f 72   /DIMM1"· ··#Memor
0070  79 2e 76 31 5f 32 31 5f  30 2e 4d 65 6d 6f 72 79   y.v1_21_ 0.Memory
0080  48 01 50 48 70 80 80 02  88 01 40 b2 01 41 0a 3f   H·PHp··· ··@··A·?
0090  2f 72 65 64 66 69 73 68  2f 76 31 2f 53 79 73 74   /redfish /v1/Syst
00a0  65 6d 73 2f 34 33 37 58  52 31 31 33 38 52 32 2f   ems/437X R1138R2/
00b0  4d 65 6d 6f 72 79 2f 44  49 4d 4d 31 2f 45 6e 76   Memory/D IMM1/Env
00c0  69 72 6f 6e 6d 65 6e 74  4d 65 74 72 69 63 73 b8   ironment Metrics·
00d0  01 03 e2 01 07 0a 05 44  49 4d 4d 31 82 02 0e 3a   ·······D IMM1···:
00e0  0c 08 00 10 01 2a 06 44  49 4d 4d 20 31 aa 02 02   ·····*·D IMM 1···
00f0  e0 5d b8 02 04 c2 02 08  08 01 10 01 18 01 20 01   ·]····· ·····
0100  ca 02 01 01 e0 02 01 8a  03 0d 0a 0b 44 49 4d 4d   ········ ····DIMM
0110  20 53 6c 6f 74 20 31 f8  03 02 ba 04 04 10 01 20    Slot 1· ·····
0120  01 12 2c 2f 72 65 64 66  69 73 68 2f 76 31 2f 53   ··,/redf ish/v1/S
```

Protobuf Wire Encoding

Results







Test Conditions:

- 24 Client Workers
- Persistent Connections

Observations:

- HTTP2 shows modest gains over HTTP1
- gRPC request rate similar across payloads
- JSON encode/decode much slower than Protobuf

HTTP1
HTTP2
gRPC

Is gRPC worth considering for Redfish?

For Redfish Clients: *Yes!*

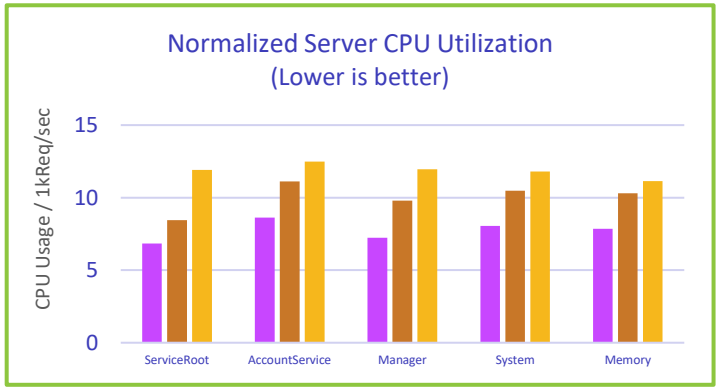
- Reduced resource usage
- Decreased network load
- Increased throughput
- Positive developer experience

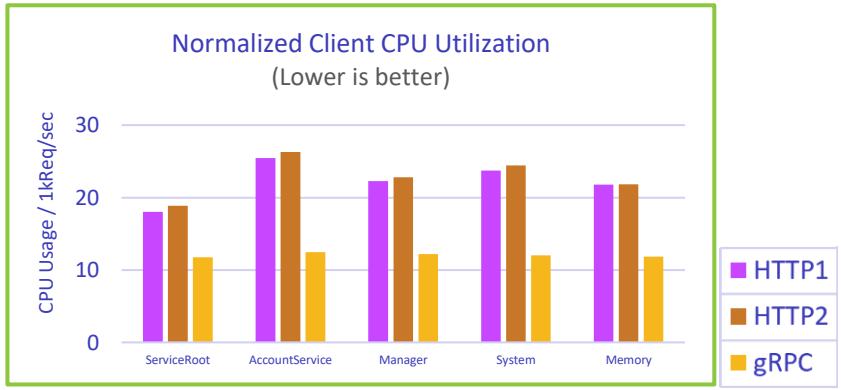
For Redfish Servers: *Maybe…*

- Increased overall throughput
- Increases load on server per client
- Strained embedded environment
  - (Protobuf Definitions Size, CPU, Marshalling)

Additional Research:

- gRPC With JSON Payload (gNMI)
- HTTP Library with Protobuf Payload
- Repeat benchmarks with Python
- Measure streaming payload



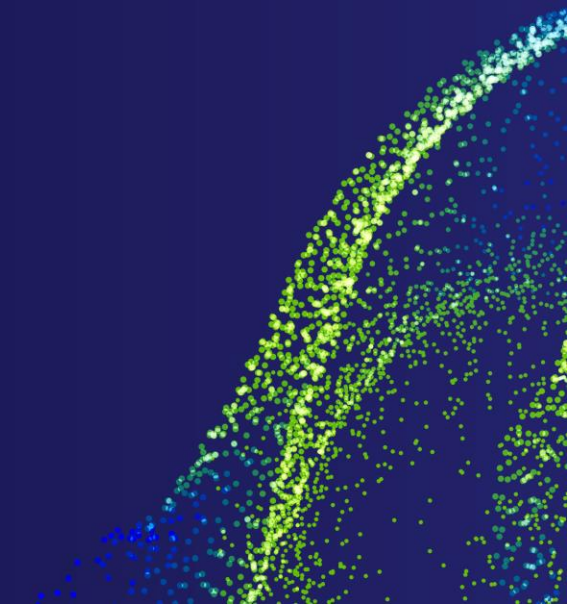


Next Steps:

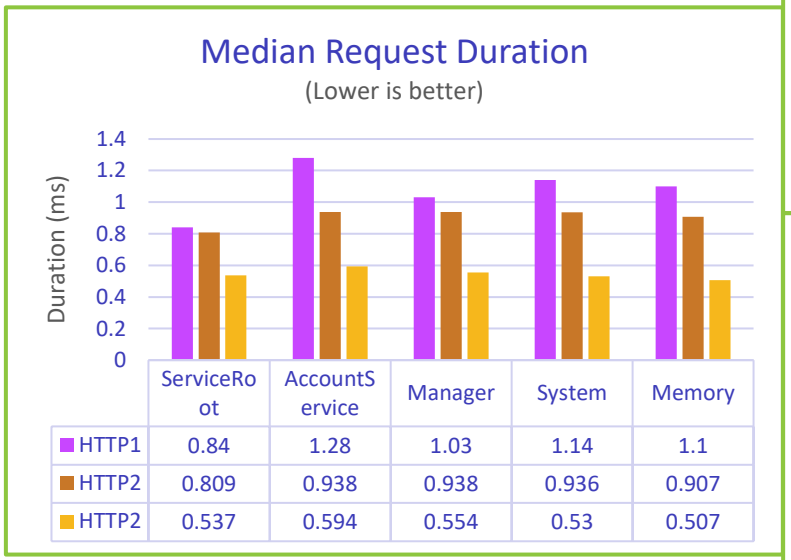
- Benchmark donated to OCP Hardware Management Project
- Protobuf conversion code donated to DMTF

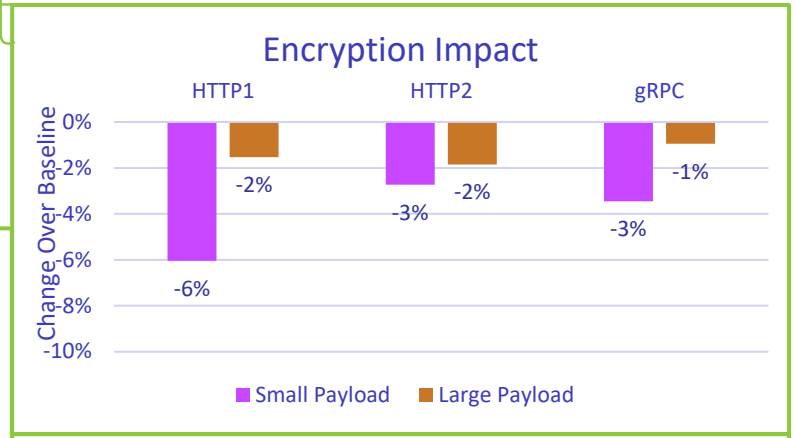
- Map additional Redfish features into RPC style
- Consider additional transport/payload configurations
- Investigate Protobuf versioning, OEM extensions

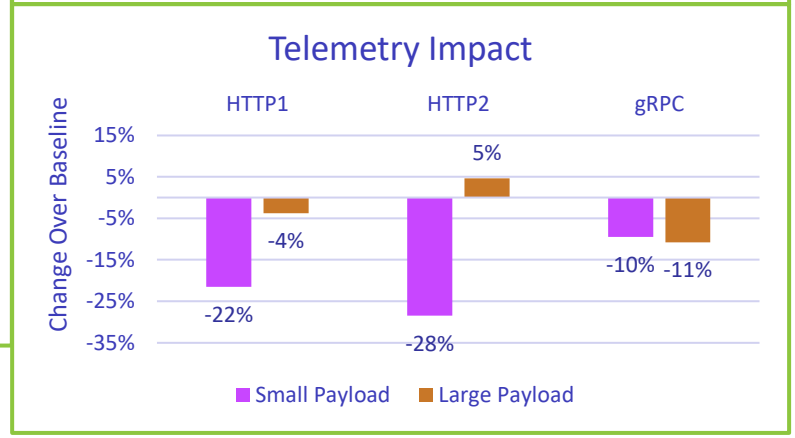
Thank You!



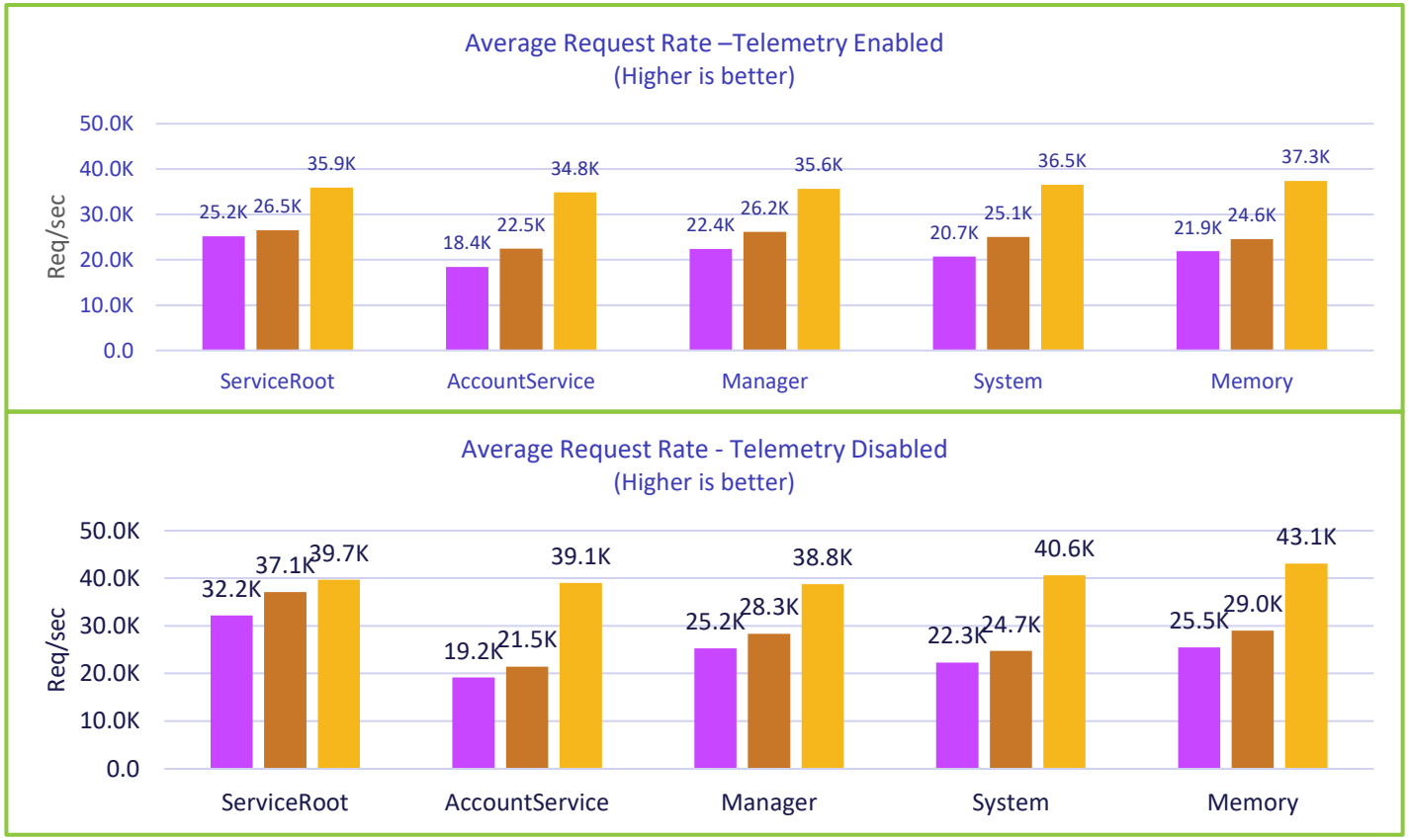
Overhead Measurement



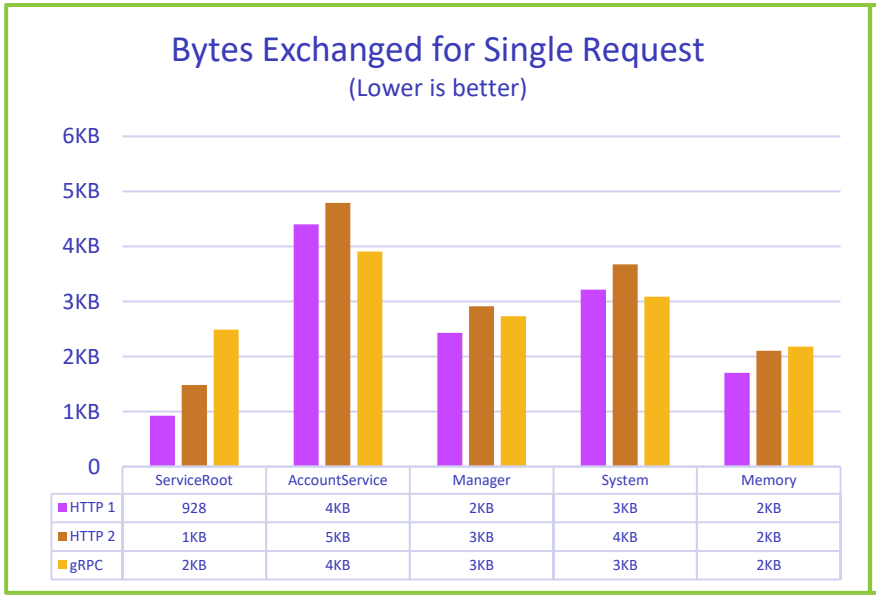




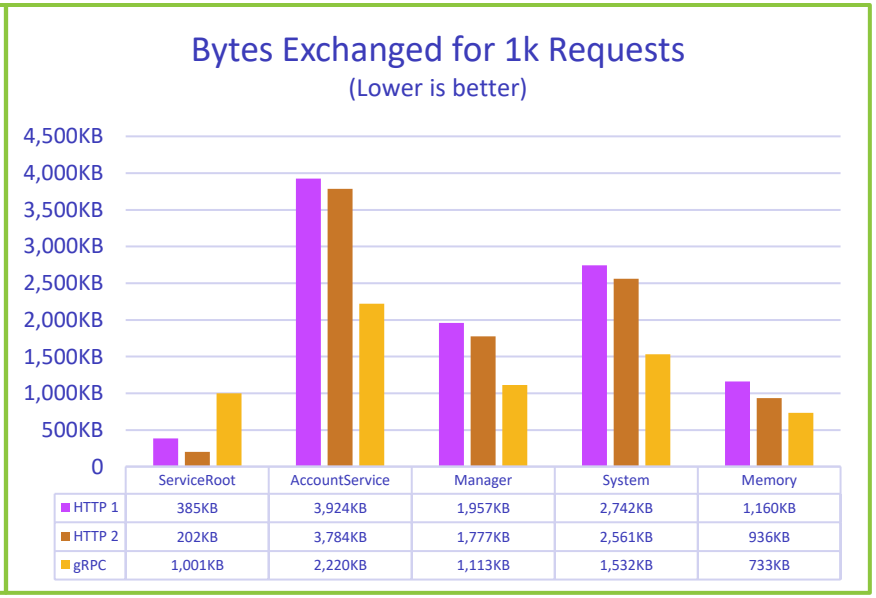
Request Rate Without Telemetry



Payload Sizes

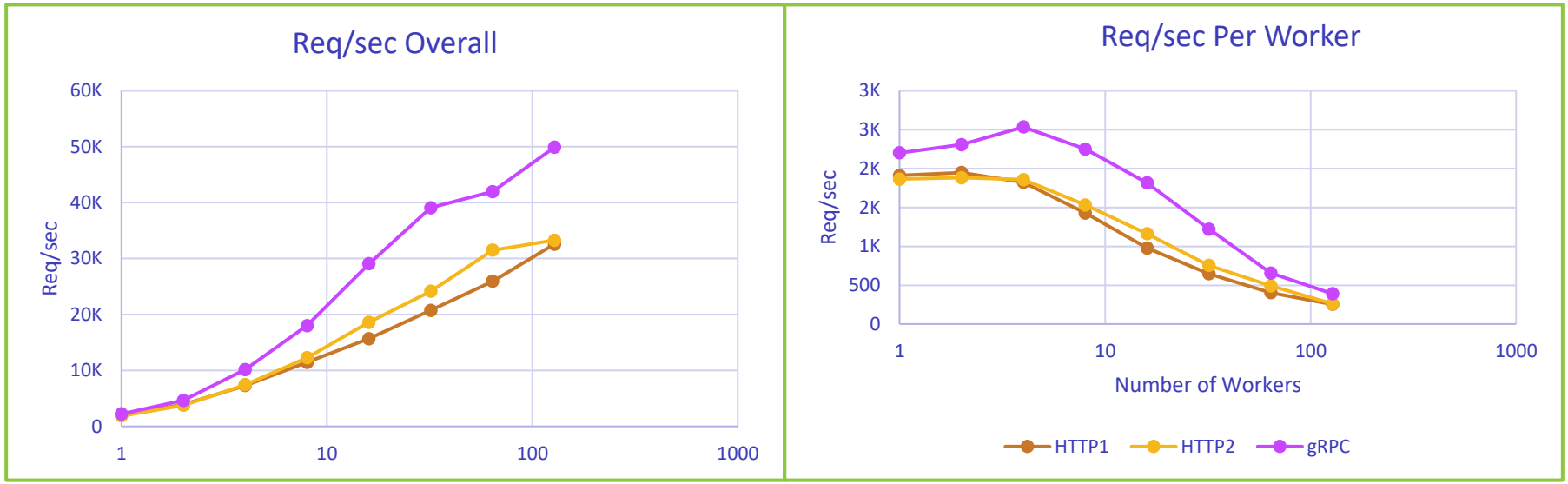


	ServiceRoot	AccountService	Manager	System	Memory
HTTP 1	928	4KB	2KB	3KB	2KB
HTTP 2	1KB	5KB	3KB	4KB	2KB
gRPC	2KB	4KB	3KB	3KB	2KB



	ServiceRoot	AccountService	Manager	System	Memory
HTTP 1	385KB	3,924KB	1,957KB	2,742KB	1,160KB
HTTP 2	202KB	3,784KB	1,777KB	2,561KB	936KB
gRPC	1,001KB	2,220KB	1,113KB	1,532KB	733KB

Worker Count



Note: Logarithmic Scale