New issue

Jump to bottom

Istio does not adhere to HTTP/2 RFC 7540 #13589

Open **trevorlinton** opened this issue on Apr 24, 2019 · 27 comments

Assignees	
Labels	area/networking lifecycle/staleproof
Projects	Prioritization
Milestone	Nebulous Future

trevorlinton commented on Apr 24, 2019 • edited

Bug description

Istio does not properly send a 421 response when a connection is reused and accident sent to a server that is not the correct origin. This can occur when there are two gateways, one with a wildcard certificate (*.example.com) and one with a different non-wildcard certificate (b.example.com) routing to two different apps (a.example.com and b.example.com) where a http/2 connection is first established to the wildcard gateway (on host a.example.com with *.example.com) then a resource is requested from an application on the non-wildcard gateway (b.example.com with certificate b.example.com).

Because of http/2 connection reuse it's possible for traffic destined for the second app (b.example.com) to end up being routed on the existing connection for (a.example.com) due to the RFC definition of connection re-use in section 9.1.1 (https://tools.ietf.org/html/rfc7540#section-9.1.1, e.g., because the certificate can authoritatively handle the request, and the IP address is the same as they are on the same ingressgateway).

When that happens, according to section 9.1.2 istio should respond with a 421 indicating that the wrong connection was used and the origin was not found. This would instruct the browser to retry on a new connection, thus renegotiating TLS and presenting SNI and thus going down the non-wildcard certificate route and to a different gateway/virtual service and the correct service.

Expected behavior

Should return a 421 and the browser should re-connect and successfully find the resource. Instead a 404 is returned.

Steps to reproduce the bug

1. Create istio 1.1.1+ instance with one ingress gateway.
2. Create a DNS record a.example.com, and b.example.com both point to the ingress gateway.
3. Create a gateway named "a" for a.example.com that uses a.example.com server host, and has a wildcard certificate for *.example.com.
4. Create a gateway named "b" for b.example.com that uses b.example.com server host and has a specific certificate b.example.com.
5. Create an app that hosts a static website with two files index.html and foobar.png. The index.html file should have an image tag that refers to an image `https://b.example.com/foobar.png` (e.g., `<img src="https://b.example.com/foobar.png">`)
6. Deploy the app twice to Kubernetes and attach virtual services for a.example.com to go to the app and b.example.com to go to the app (effectively a.example.com and b.example.com are both hosting the app with different certificates on the same IP address, where a is on a wildcard cert and b is not).
7. Visit https://a.example.com/, notice that you receive a 404 in Chrome and Firefox but not safari or opera.

Version (include the output of `istioctl version --remote` and `kubectl version`)

```
kubectl vclient version: version.BuildInfo{Version:"1.1.1", GitRevision:"2b1331886076df103179e3da5dc9077fed59c989", User:"root", Host:"7077232d-4c6c-11e9-813c-0a580a2c0506",
GolangVersion:"go1.10.4", DockerHub:"docker.io/istio", BuildStatus:"Clean", GitTag:"1.1.0-17-g2b13318"}
apps-private-ingressgateway version: version.BuildInfo{Version:"", GitRevision:"", User:"", Host:"", GolangVersion:"", DockerHub:"", BuildStatus:"", GitTag:""}
apps-public-ingressgateway version: version.BuildInfo{Version:"", GitRevision:"", User:"", Host:"", GolangVersion:"", DockerHub:"", BuildStatus:"", GitTag:""}
citadel version: version.BuildInfo{Version:"1.1.2", GitRevision:"2b1331886076df103179e3da5dc9077fed59c989-dirty", User:"root", Host:"35adf5bb-5570-11e9-b00d-0a580a2c0205",
GolangVersion:"go1.10.4", DockerHub:"docker.io/istio", BuildStatus:"Modified", GitTag:"1.1.1"}
galley version: version.BuildInfo{Version:"1.1.2", GitRevision:"2b1331886076df103179e3da5dc9077fed59c989-dirty", User:"root", Host:"35adf5bb-5570-11e9-b00d-0a580a2c0205",
GolangVersion:"go1.10.4", DockerHub:"docker.io/istio", BuildStatus:"Modified", GitTag:"1.1.1"}
pilot version: version.BuildInfo{Version:"1.1.2", GitRevision:"2b1331886076df103179e3da5dc9077fed59c989-dirty", User:"root", Host:"35adf5bb-5570-11e9-b00d-0a580a2c0205",
GolangVersion:"go1.10.4", DockerHub:"docker.io/istio", BuildStatus:"Modified", GitTag:"1.1.1"}
policy version: version.BuildInfo{Version:"1.1.2", GitRevision:"2b1331886076df103179e3da5dc9077fed59c989-dirty", User:"root", Host:"35adf5bb-5570-11e9-b00d-0a580a2c0205",
GolangVersion:"go1.10.4", DockerHub:"docker.io/istio", BuildStatus:"Modified", GitTag:"1.1.1"}
sidecar-injector version: version.BuildInfo{Version:"1.1.2", GitRevision:"2b1331886076df103179e3da5dc9077fed59c989-dirty", User:"root", Host:"35adf5bb-5570-11e9-b00d-0a580a2c0205",
GolangVersion:"go1.10.4", DockerHub:"docker.io/istio", BuildStatus:"Modified", GitTag:"1.1.1"}
telemetry version: version.BuildInfo{Version:"1.1.2", GitRevision:"2b1331886076df103179e3da5dc9077fed59c989-dirty", User:"root", Host:"35adf5bb-5570-11e9-b00d-0a580a2c0205",
GolangVersion:"go1.10.4", DockerHub:"docker.io/istio", BuildStatus:"Modified", GitTag:"1.1.1"}
sites-private-ingressgateway version: version.BuildInfo{Version:"", GitRevision:"", User:"", Host:"", GolangVersion:"", DockerHub:"", BuildStatus:"", GitTag:""}
sites-public-ingressgateway version: version.BuildInfo{Version:"", GitRevision:"", User:"", Host:"", GolangVersion:"", DockerHub:"", BuildStatus:"", GitTag:""}

$ kubectl version
Client Version: version.Info{Major:"1", Minor:"10", GitVersion:"v1.10.4", GitCommit:"5ca598b4ba5abb89bb773071ce452e33fb66339d", GitTreeState:"clean", BuildDate:"2018-06-
18T14:14:00Z", GoVersion:"go1.9.7", Compiler:"gc", Platform:"darwin/amd64"}
Server Version: version.Info{Major:"1", Minor:"11", GitVersion:"v1.11.5", GitCommit:"753b2dbc622f5cc417845f0ff8a77f539a4213ea", GitTreeState:"clean", BuildDate:"2018-11-
26T14:31:35Z", GoVersion:"go1.10.3", Compiler:"gc", Platform:"linux/amd64"}
```

How was Istio installed?

Helm

Environment where bug was observed (cloud vendor, OS, etc)

AWS, with istio installed and running with an NLB ingress or as a nodeport terminating the TLS.

👍 11

trevorlinton commented on Apr 24, 2019

Author

Seems related: #9429

howardjohn added the area/networking label on Apr 24, 2019

howardjohn commented on Apr 24, 2019 Member

> This status code MUST NOT be generated by proxies.

Seems like the spec says the we cannot return a 421? Or maybe I am reading that wrong

trevorlinton commented on Apr 24, 2019 • edited Author

@howardjohn they mean proxies in the traditional sense, istio could be considered a reverse proxy but its not a proxy by the RFC definition. Folks at chromium also clarified that the proxy only applies to forward proxies: https://bugs.chromium.org/p/chromium/issues/detail?id=954160#c5 (see second to last comment)

trevorlinton commented on May 1, 2019 Author

@howardjohn thoughts?

howardjohn commented on May 1, 2019 Member

I agree with what you said, Istio isn't a proxy by the rfc definition

Seems like a bug but probably something to change in Envoy rather than Istio?

trevorlinton commented on May 1, 2019 Author

I can open an issue on envoy, link the tickets, we'll see where we land, for now if its alright to leave this ticket up to let other teams comment that would be helpful.

trevorlinton mentioned this issue on May 1, 2019

Envoy does not adhere to HTTP/2 RFC 7540 envoyproxy/envoy#6767
Open

trevorlinton commented on May 1, 2019 Author

@howardjohn If you'd like to expand what envoy would need to fix it would help, I've opened a ticket for them here: envoyproxy/envoy#6767

PiotrSikora commented on May 2, 2019 Contributor

@trevorlinton could you share `/config_dump` from Envoy?

trevorlinton commented on May 2, 2019 Author

@PiotrSikora I'm unsure how to grab that, I'm not terribly familiar with envoy, just as much as exposed by istio. Here's the `istioctl proxy-config routes` if it helps, I can directly jump on the envoy pod if you know the local port to get that information.

```
[
    {
        "name": "https.443.https-apps-public.apps-public.sites-system",
        "virtualHosts": [
            {
                "name": "appa.example.com:443",
                "domains": [
                    "appa.example.com",
                    "appa.example.com:443"
                ],
                "routes": [
                    {
                        "match": {
                            "prefix": "/"
                        },
                        "route": {
                            "cluster": "outbound|80||appa.default.svc.cluster.local",
                            "timeout": "0s",
                            "retryPolicy": {
                                "retryOn": "connect-failure,refused-stream,unavailable,cancelled,resource-exhausted,retriable-status-codes",
                                "numRetries": 2,
                                "retryHostPredicate": [
                                    {
                                        "name": "envoy.retry_host_predicates.previous_hosts"
                                    }
                                ],
                                "hostSelectionRetryMaxAttempts": "3",
                                "retriableStatusCodes": [
                                    503
                                ]
                            },
                            "maxGrpcTimeout": "0s"
                        },
                        "metadata": {
                            "filterMetadata": {
                                "istio": {
                                    "config": "/apis/networking/v1alpha3/namespaces/sites-system/virtual-service/appa-default"
                                }
                            }
                        },
                        "decorator": {
                            "operation": "appa.default.svc.cluster.local:80/*"
                        },
                        "perFilterConfig": {
                            "mixer": {
                                "forward_attributes": {
                                    "attributes": {
                                        "destination.service.host": {
                                            "string_value": "appa.default.svc.cluster.local"
                                        },
                                        "destination.service.name": {
                                            "string_value": "appa"
                                        },
                                        "destination.service.namespace": {
                                            "string_value": "default"
                                        },
                                        "destination.service.uid": {
                                            "string_value": "istio://default/services/appa"
                                        }
                                    }
                                },
                                "mixer_attributes": {
                                    "attributes": {
                                        "destination.service.host": {
                                            "string_value": "appa.default.svc.cluster.local"
                                        },
                                        "destination.service.name": {
                                            "string_value": "appa"
                                        },
                                        "destination.service.namespace": {
                                            "string_value": "default"
                                        },
                                        "destination.service.uid": {
                                            "string_value": "istio://default/services/appa"
                                        }
                                    }
                                }
                            }
                        }
                    }
                ]
            }
        ],
        "validateClusters": false
    },
    {
        "name": "https.443.https-appb-example-com.appb-public.sites-system",
        "virtualHosts": [
            {
                "name": "appb.example.com:443",
                "domains": [
                    "appb.example.com",
                    "appb.example.com:443"
                ],
                "routes": [
                    {
                        "match": {
                            "prefix": "/"
                        },
                        "route": {
                            "cluster": "outbound|80||appb.default.svc.cluster.local",
                            "timeout": "0s",
                            "retryPolicy": {
                                "retryOn": "connect-failure,refused-stream,unavailable,cancelled,resource-exhausted,retriable-status-codes",
                                "numRetries": 2,
                                "retryHostPredicate": [
                                    {
                                        "name": "envoy.retry_host_predicates.previous_hosts"
                                    }
                                ],
                                "hostSelectionRetryMaxAttempts": "3",
                                "retriableStatusCodes": [
                                    503
                                ]
                            },
                            "maxGrpcTimeout": "0s"
                        },
                        "metadata": {
                            "filterMetadata": {
                                "istio": {
                                    "config": "/apis/networking/v1alpha3/namespaces/sites-system/virtual-service/appb-default"
                                }
                            }
                        },
                        "decorator": {
                            "operation": "appb.default.svc.cluster.local:80/*"
                        },
                        "perFilterConfig": {
                            "mixer": {
                                "forward_attributes": {
                                    "attributes": {
                                        "destination.service.host": {
                                            "string_value": "appb.default.svc.cluster.local"
                                        },
                                        "destination.service.name": {
                                            "string_value": "appb"
                                        },
                                        "destination.service.namespace": {
                                            "string_value": "default"
                                        },
                                        "destination.service.uid": {
                                            "string_value": "istio://default/services/appb"
                                        }
                                    }
                                },
                                "mixer_attributes": {
                                    "attributes": {
                                        "destination.service.host": {
                                            "string_value": "appb.default.svc.cluster.local"
                                        },
                                        "destination.service.name": {
                                            "string_value": "appb"
                                        },
                                        "destination.service.namespace": {
                                            "string_value": "default"
                                        },
                                        "destination.service.uid": {
                                            "string_value": "istio://default/services/appb"
                                        }
                                    }
                                }
                            }
                        }
                    }
                ]
            }
        ],
        "validateClusters": false
    },
]
```

PiotrSikora commented on May 2, 2019 Contributor

@trevorlinton it's `localhost:15000/config_dump`, but I think `istioctl proxy-config` returns the same information (I keep forgetting about it...).

Could you also paste output from `istioctl proxy-config listeners`?

trevorlinton commented on May 6, 2019 Author

Sadly I removed our test cluster after I dumped the first configuration, I can give you instructions in Istio as to how to re-create it.

PiotrSikora commented on May 7, 2019 Contributor

@trevorlinton that would be great, thanks.

trevorlinton commented on May 9, 2019 Author

1. Create one IP address or ingress-gateway in istio
2. Create two domain entries a.example.com and b.example.com both pointing to the same IP address and ingress-gateway
3. Create two certificates one which is a wildcard *.example.com and one which is a regular single certificate without a sans for b.example.com
4. Create two applications the first should host a.example.com and only serve up a static index.html which references an image called foo.png (via an img tag) from https://b.example.com/foo.png. The second application should be host b.example.com and only host foo.png.
5. Create two gateways one for a.example.com and one for b.example.com, the a.example.com should use the wildcard certificate, b.example.com should use the certificate b.example.com from step 3.
6. Create two virtual services one for a.example.com that uses gateway a.example.com and points ot the application for a.example.com and one for b.example.com should likewise use gateway b.example.com and points to the application for b.example.com.

Then visit a.example.com on Chrome or Firefox. Notice how the reference for b.example.com/foo.png returns a 404. Where the expected result is it should return a 421, the browser should re-negotiate the SNI and a new connection then the image should eventually be resolved and the site should successfully work. Notice that via curl you can retrieve both https://a.example.com/index.html and https://b.example.com/foo.png but in the browser going to https://b.example.com/foo.png fails (even when directly navigating to it) if you've first been to a.example.com and have it open in a different tab.

Hope this helps replicate it.

andraxylia assigned **PiotrSikora** on Jun 27, 2019

andraxylia added this to the **Nebulous Future** milestone on Jun 27, 2019

alexbrand mentioned this issue on Sep 12, 2019

Requests can be misrouted due to HTTP/2 Connection Coalescing under certain scenarios projectcontour/contour#1493

Closed

stale bot commented on Sep 26, 2019

This issue has been automatically marked as stale because it has not had activity in the last 90 days. It will be closed in the next 30 days unless it is tagged "help wanted" or other activity occurs. Thank you for your contributions.

stale bot added the **stale** label on Sep 26, 2019

stale bot commented on Oct 26, 2019

This issue has been automatically closed because it has not had activity in the last month and a half. If this issue is still valid, please ping a maintainer and ask them to label it as "help wanted". Thank you for your contributions.

stale bot closed this as completed on Oct 26, 2019

istio-policy-bot removed the **stale** label on Oct 26, 2019

haf commented on Apr 14, 2020 • edited

This issue should be reopened with an external ref envoyproxy/envoy#6767

> In some deployments, reusing a connection for multiple origins can result in requests being directed to the wrong origin server. For example, TLS termination might be performed by a middlebox that uses the TLS Server Name Indication (SNI) [TLS-EXT] extension to select an origin server. This means that it is possible for clients to send confidential information to servers that might not be the intended target for the request, even though the server is otherwise authoritative.

> **A server that does not wish clients to reuse connections can indicate that it is not authoritative for a request by sending a 421 (Misdirected Request) status code in response to the request (see Section 9.1.2).**

— https://httpwg.org/specs/rfc7540.html#reuse

@istio/wg-networking-maintainers

trevorlinton commented on Apr 15, 2020 Author

A CVE was opened (not by me) surrounding this issue: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-11767

craigbox commented on Apr 20, 2020 Contributor

/reopen

craigbox reopened this on Apr 20, 2020

howardjohn mentioned this issue on May 13, 2020

404 NR when using browser on multiple ingress gateways #9429

Closed

istio-policy-bot added the lifecycle/stale label on Jul 20, 2020

nc-ruth commented on Jul 20, 2020

I still think this is relevant, and should not be closed, thank you.

craigbox added lifecycle/staleproof and removed lifecycle/stale labels on Jul 20, 2020

howardjohn mentioned this issue on Sep 23, 2020

[istio-gateway] with both PASSTHROUGH and TLS servers will 404 PASSTHROUGH hosts depending on access order from client. #26972

Closed

howardjohn commented on Oct 7, 2020 Member

How other proxies handle this:

Apache: https://httpd.apache.org/docs/2.4/mod/mod_http2.html#misdirected automatic detection of a vhost requested that has different TLS config
Contour (envoy): https://github.com/projectcontour/contour/pull/2483/files#diff-98b92a08d0022a6c73eceeb2e1d99a43R303. This is a lua filter that asserts the SNI matches the host header
Traefik: traefik/traefik#7008 adds option (enabled by default) to assert SNI matches host header
Nginx: https://forum.nginx.org/read.php?29,267026 not sure I fully understood this one, but looks like it is asserting SNI matches Host

https://tools.ietf.org/html/rfc6066#section-11.1 seems to imply we should be asserting SNI matches Host

howardjohn added this to **P1** in **Prioritization** on Oct 23, 2020

howardjohn mentioned this issue on Nov 30, 2020

Nondeterministic behaviour of istio-ingressgateway with mTLS #29214

Closed

howardjohn mentioned this issue on Mar 5, 2021

Client Certificate Verification for Gateway Listeners kubernetes-sigs/gateway-api#91

Open

FrimIdan commented on Mar 11, 2021 Contributor

Hi,

Do we have a WA for this issue? I've facing the exact issue with wildcard cert and 3 different services

Boojapho commented on Apr 15, 2021

> Do we have a WA for this issue? I've facing the exact issue with wildcard cert and 3 different services

There are three choices to work around the issue. These focus on making sure the browser doesn't try to reuse your connections:

- Create non-overlapping certificates for each gateway. You can use subdomains for each gateway if you still want wildcard certs.
- Create a different proxy port for each gateway on the front-end. For example use port 443 for main apps and 8443 for admin apps.

- Create proxies (e.g. load balancers) with different external IPs for each gateway

The browser reuses connections if the IP:Port is the same as a previous connection and the previous connection's cert is valid for the new hostname. See the HTTP spec for details. If you can affect one of those 3 items (IP, Port, Cert), you can avoid the connection reuse, which works around Istio's limitation.

howardjohn mentioned this issue on Apr 19, 2021

Ingress gateway access log - wrong requested_server_name #32271

Closed

desimone mentioned this issue on Apr 28, 2021

safari does not respect `421` status for HTTP/2 connection reuse / coalescing pomerium/pomerium#2150

Closed

PiotrSikora assigned **lambdai** and unassigned **PiotrSikora** on May 15, 2021

ragingpastry commented on Sep 16, 2021

We ran into this issue and ended up disabling HTTP/2 in istio with the following config.

```
apiVersion: networking.istio.io/v1alpha3
kind: EnvoyFilter
metadata:
  name: disable-alpn-h2
  namespace: istio-system
spec:
  workloadSelector:
    labels:
      istio: ingressgateway
  configPatches:
  - applyTo: FILTER_CHAIN
    match:
      listener:
        filterChain:
          sni: "*.mygateway.com"
    patch:
      operation: MERGE
      value:
        transportSocket:
          name: envoy.transport_sockets.tls
          typedConfig:
            '@type': type.googleapis.com/envoy.extensions.transport_sockets.tls.v3.DownstreamTlsContext
            commonTlsContext:
              alpnProtocols:
                - "http/1.1"
              tlsCertificateSdsSecretConfigs:
                - name: kubernetes://wildcard-cert
                  sdsConfig:
                    ads: {}
                    resourceApiVersion: V3
```

jiangshantao-dbg mentioned this issue on Nov 22, 2021

feat: aLPN only use http1.1 by default istio-mt/istio#8

Merged

jiangshantao-dbg commented on Dec 28, 2021 • edited

> We ran into this issue and ended up disabling HTTP/2 in istio with the following config.
>
> ```
> apiVersion: networking.istio.io/v1alpha3
> kind: EnvoyFilter
> metadata:
>   name: disable-alpn-h2
>   namespace: istio-system
> spec:
>   workloadSelector:
>     labels:
>       istio: ingressgateway
>   configPatches:
>   - applyTo: FILTER_CHAIN
>     match:
>       listener:
>         filterChain:
>           sni: "*.mygateway.com"
>     patch:
>       operation: MERGE
>       value:
>         transportSocket:
>           name: envoy.transport_sockets.tls
>           typedConfig:
>             '@type': type.googleapis.com/envoy.extensions.transport_sockets.tls.v3.DownstreamTlsContext
>             commonTlsContext:
>               alpnProtocols:
>                 - "http/1.1"
>               tlsCertificateSdsSecretConfigs:
>                 - name: kubernetes://wildcard-cert
>                   sdsConfig:
>                     ads: {}
>                     resourceApiVersion: V3
> ```

@ragingpastry this not work for me. cause `protoMerge` to array `alpnProtocols` with appending action not replacing.
we have to modify the gateway listener builder, `pilot/pkg/networking/core/v1alpha3/gateway.go`

```
        ctx := &tls.DownstreamTlsContext{
                  CommonTlsContext: &tls.CommonTlsContext{
  -                       AlpnProtocols: util.ALPNHttp,
  +                       AlpnProtocols: util.ALPNHttpOnly,
                  },
          }
```

@howardjohn can we expose the AlpnProtocols config to `Gateway.Servers[].Tls.AlpnProtocols` API.
we disable the http2 by default. when app need http2 use EnvoyFilter to turn on the http2 support.
but this cause server prefers to use http1.1, because http1.1 is in front of http2.

```
name: envoy.transport_sockets.tls
typed_config:
  '@type': >-
    type.googleapis.com/envoy.extensions.transport_sockets.tls.v3.DownstreamTlsContext
  common_tls_context:
    alpn_protocols:
      - http/1.1
      - h2
    tls_certificate_sds_secret_configs:
      - name: kubernetes://xxx
        sds_config:
          ads: {}
          resource_api_version: V3
  require_client_certificate: false
```

```
## SSL ALPN logic
SSL_select_next_proto() is a helper function used to select protocols. It implements the standard protocol selection.
It is expected that this function is called from the application callback cb.
The protocol data in server, server_len and client, client_len must be in the protocol-list format described below.
The first item in the server, server_len list that matches an item in the client, client_len list is selected, and returned in out, outlen.
The out value will point into either server or client, so it should be copied immediately.
If no match is found, the first item in client, client_len is returned in out, outlen.
This function can also be used in the NPN callback.
```

```
if (!parsed_alpn_protocols_.empty() && !config.capabilities().handles_alpn_selection) {
      SSL_CTX_set_alpn_select_cb(
          ctx.ssl_ctx_.get(),
          [](SSL*, const unsigned char** out, unsigned char* outlen, const unsigned char* in,
             unsigned int inlen, void* arg) -> int {
            return static_cast<ServerContextImpl*>(arg)->alpnSelectCallback(out, outlen, in, inlen);
          },
          this);
    }
```

```
int ServerContextImpl::alpnSelectCallback(const unsigned char** out, unsigned char* outlen,
                                          const unsigned char* in, unsigned int inlen) {
  // Currently this uses the standard selection algorithm in priority order.
  const uint8_t* alpn_data = parsed_alpn_protocols_.data();
  size_t alpn_data_size = parsed_alpn_protocols_.size();

  if (SSL_select_next_proto(const_cast<unsigned char**>(out), outlen, alpn_data, alpn_data_size, in,
                            inlen) != OPENSSL_NPN_NEGOTIATED) {
    return SSL_TLSEXT_ERR_NOACK;
  } else {
    return SSL_TLSEXT_ERR_OK;
  }
}
```

```
parsed_alpn_protocols_ = parseAlpnProtocols(config.alpnProtocols());
```

now we have to reorder the AlpnProtocols fields after the listener patches, this is so tricky way.

Look forward to your reply.

thanks! @howardjohn

jiangshantao-dbg mentioned this issue on Dec 28, 2021

feat: order the alpn protocol to prefer to use h2 istio-mt/istio#11

Merged

craigbox commented on Jan 23 Contributor

Perhaps someone could take this issue to a Networking WG meeting?

howardjohn commented on Jan 23 Member

This has been discussed a few times in the past and we were unable to achieve consensus on a path forward. I think **@lambdai** has a doc about this we can probably attach for more context

jingslunt commented on May 25

> This has been discussed a few times in the past and we were unable to achieve consensus on a path forward. I think **@lambdai** has a doc about this we can probably attach for more context

how can I tell the chrome disable http2 such as "chrome --disable-http2" , or tell the istio gateway protocol: HTTPS1.1

Assignees

lambdai

Labels

area/networking lifecycle/staleproof

Projects

Prioritization
P1

Milestone

Nebulous Future

Development

No branches or pull requests

14 participants