HAPPY
Internet-Draft
Intended status: Informational
Expires: 18 May 2026

J. Palet Martinez
The IPv6 Company
P. S. Tiesel
SAP SE

14 November 2025

Considerations for Happy Eyeballs Error Reporting
draft-palet-happy-reporting-considerations-00

Abstract

This document introduces diferent aspects to be considered for the Happy Eyeballs error reporting.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 18 May 2026.

Copyright Notice

Table of Contents

1. Introduction

Happy Eyeballs ([I-D.ietf-happy-happyeyeballs-v3]) provides a way for improving user-visible delay when FQDN's have multiple IP addresses and connectivity is performing worse in those that should be the preferred ones.

However, this hides possible connectivity issues to the operator or other parties in the chain between the client and the service being accesed, because users will not notice anything broken, so they will not report it to the providers. For example, in the case of a dual-stack web site, if IPv6 connectivity is somehow broken at any point between the client and the hosting service, Happy Eyeballs (HE across the rest of this document), will quicly fall-back to IPv4.

The goal of this document is to discuss diferent aspects to be considered, in order to provide a decision path towards the best possible choice for the final error reporting solution for HE. The error reporting solution should allow an integral HE error reporting mechanism that enables setting up alarms and triggering further investigations so to improve network reliability, facilitating the detection of failures as soon as they appear, without the need of aditional external monitoring.

2. Use cases

In general, there are basically 4 possible use cases or parts of the network that can cause HE fallback (IPv6 to IPv4 or QUIC to TCP):

1. Customer internal network. It may be a managed (typically an enterprise network, but it may be also a residential subscriber or SME network) or unmanaged network (most common case of

residential subscribers or SME networks). In some cases the CE is also managed by the customer, while in most of the cases it is managed by the service provider (at least the WAN side). This is a subscriber problem, typically caused by some missconfiguration (e.g., inadequate filtering at firewalls, switches or routers) and most of the time, the most interested party for the reporting is the subscriber itself, but up to a certain point, it is also interesting for the service provider (specially if the problem is caused by CE missconfiguration). It is also interesting for the service provier to know when a fallback problem exists, even inside of the customer network, because typically the failures may reduce performance, or increase the usage of more expensive network resources for the service provider network (for instance IPv4 instead of IPv6).

2. Service provider network. This is typically caused by some problem at the WAN side of the subscriber CE or in other parts of the service provider network, including the direct upstreams (peerings/transits). The most interested party for the reporting of the failures is the service provider itself. For example in case it is a misconfiguration of the CE itself, if managed by the service provider, it can be easily fixed, once the service provider is notified about the problem. Similarly if the problem is in other parts of the service provider network (e.g. a temporary failure of IPv6 in the upstreams that hasn’t been notified by other management systems). In the case of cellular networks, if still using dual-stack instead of IPv6-only, most of the time the problem will be located in the service provider network, not in the internal customer network, neither in the UE. In all those cases, the service provider can directly fix any issues or if they are created by the direct upstreams notify to them so they are resolved.

3. Intermediate transits. This reporting is also interesting for the service provider, however will not be able to resolve it direclty and will need to forward the case to the relevant upstream(s), so they scale it to the relevant parties. In fact the service provider may be unable to know exactly where the problem is being created. Intermediate transits will not have any interest in being direct parties for the error reporting. "Human communication" will be typically more efficient, such as by means of the network operator groups, checking with other service providers sharing the same transits, etc.

4. Content provider. This has been the most common source of the problem for some time, for instance, content providers having configured AAAA RR’s when IPv6 connectivity was not good or even inexistent, wrong configuration in load-balancers or firewalls,

etc. This is clearly more interesting for the content provider. If the service provider gets the relevant info, they can try to contact the content provider ("human communication" by network operator grups works also here), as it may happen that this is happening not just for a single subcriber.

5. Application developers. To validate their own applications and infrastructure as well as providing customers support, application developers may want to be able to trace why certain communication prefer IPv4 instead of IPv6. In this case, it is especially important to understand whether the client has received an AAAA RR, whether connecting IPv6 was actually tried (and potentially why not) and how the attempt failed, e.g., lost race, TCP/TLS/QUIC handshake failed, … Especially for web applications, exposing this information through developer tools or performance logs is crucial.

If the error reporting is sent to the content provider, they will only be able to fix it if it is a general problem, affecting any possible source address in the Internet. However, they will not be able to test "more specific" cases. They will be unable to understand the problem or even fix it, if it is caused by 1, 2 or 3 above.

Instead, if the reporting is sent to the service provider, he is most probably able to verify it and initiate a resolution path for:

* Case 1, either by fixing the CE configuration or by notifying the subscriber about their missconfiguration.

* Case 2, either by the required network changes or if needed by contacting its upstreams.

* Case 3, either by contacting their upstreams so they can forward it up in the chain, or by using network operator groups contacts.

* Case 4, either contacting the content provider or by means of network operator groups.

In all those cases, the service provider is also able to re-verify the problem and if needed cooperate with other involved parties.

In the case 5, for developers, some kind of plug-in for the developer tools and performance logs is needed in order to understand HE decisions. Same is true for language libraries that do happy eyeballs under the hood. So in this case, it may be something that can be turned on/off by the developer code, as part of tracing/ debugging facilities.

3. What to report?

TBD: Discussion needed.

In order to be able to avoid the fallback, it seems that source address (possibly just a prefix, not individual address, which also may resolve privacy issues), destination address (or even FQDN) are needed. It seems logic also to inform about what destination address failed, which one succeded, and if the problem was IPv6 (fallback to IPv4) or QUIC (fallback to TCP). It seems also convenient to inform about the timers that caused the fallback (Resolution Delay, Connection Attempt Delay)?.

4. When to report?

TBD: Discussion needed.

Should the report be sent with every failure, or should we determine a minimum number of failures from a client to a given destination before reporting? Temporary failures may be not relevant, but also may be an indication of some kind of network flapping.

5. How to report?

TBD: Discussion needed.

If we decide that the reporting should be done only to the service provider, then experience shows that they are not supportive of implementing anything new. In any case, we could consider several ways to do the reporting:

a. Existing mechanism widely used by service/content providers (example syslog). Another option could be using the system default logging channel (windows event logs, systemd-journal, etc.).

b. New mechanism to be used by content providers (W3C Network Error Logging [NEL]).

c. New mechanism, designed on purpose for HE error reporting (such as ICMP, [I-D.trammell-happy-sad]).

d. Other choices, even multiple reporting mechanisms for different use cases?.

It seems that W3C Network Error Logging [NEL] will be only used by content providers, not clear if wil get "universal" deployment, and as we already argued above, it looks like the best candidate for a higher impact in most of the use cases, it is reporting to the service provider.

Choosing a new mechanism, means it must be deployed on purpose. As said before, service providers aren’t typically happy implementing anything new.

If we choose a mechanism which is widely adopted by service providers it is clear that still the sevice provider need to configure something new in the network. For example, in the case of syslog ([RFC5424]) over UDP ([RFC5426]), widely used by service and content providers, the "listening service" is already implemented, but as for any new type of alerts, some additional ticketing or procedures should be setup. However, this will be the same for any kind of reporting mechanism.

Using the system default logging channel is something that may need some exploration. In a first sight, it seems to imply different logging solutions on the side of the service provider or data-center, but it can be interesting for the case of reporting in enterprise networks. The open question is if it makes sense to have multiple reporting protocols for different cases, and how much this could increase the complexity for the HE developers itself.

Considerations for choosing a protocol: Balance of work to be done in reporting hosts vs service provider. Chances to be implemented in hosts vs chances to be implemented in service providers. TBD.

Format: JSON, QLOG? TBD.

Service discovery to identify the listener of the reporting protocols: IANA dual-stack defined address? TBD.

6. Privacy Considerations

TBD. Very draft text follows from previous work and list inputs.

The goal is to provide the operator information about the failures detected by HE, without requiring specific users traffic information. Towards this, it will be sufficient to provide to the error collector details about the failed destination address and source prefix. So privacy issues regarding identification of a specific device or users are avoided.

Nowadays, operators already log this information in order to comply with lawful interception regulations, and in general, data protection regulations allow this logging when technically required. Data protection regulations explicitly say that the data can't be disclosed, and there is no need to do so.

In general, vendors also collect telemetry data from devices, in order to improve OSs and in some situations, there are regulations that enforce offering the user to enable/disable that feature. So we could consider offering the same feature for this mechanism.

When the mechanism described in this document detects a failure, the operator will need to find if the problem is related to:

* A specific subscriber (customer internal networks, or even at their CE).

* A group of subscribers or the entire service provider network (e.g., one or several part service provider network).

* Intermediate transits.

* Content provider.

Those cases, in terms of privacy considerations, will fall into one of the following categories:

a. Failure cause in customer internal network: The operator may decide, depending on their country regulations and services offered to that customer, to inform the customer (and decide what information is provided), or ignore the failure and include it in a "while list" (i.e., list of "don't care" failures), so the monitoring system doesn't keep providing alerts on it.

b. Failure cause due to the service provider network: The operator will need to find the cause and fix the failure, without disclosing any personal data.

c. Failure cause due to third parties (intermediate transits or content provider): The operator don't need to disclose any specific user source address/prefix, because in this case, the shorter prefix (typically the RIR allocated prefix or part of it, when is being announced split among different BGP peers), from which the failure has been verified is sufficient to re-verify the error.

In the most extreme case, a more restrictive usage of this procedure, not involving logging any user source address/prefix, will be to log only the failed destination address. In a big percentage of the cases, it will be enough for the service provider to detect the failure (use cases 2, 3, and 4), as experience shows that HE fallback occurs mainly because path or destination misconfiguration or issues. So, the service provider could replicate the failure from any other source address in its network to the same failed destination. If we take this approach, failures internal to a specific subscriber, could not be reported by the operator to the customer (as there is no source data logging), and together with partial failures of the operator network will require extra work from operator's staff to research the cause of the failure (i.e., it is in my network, part of it, a specific customer or external).

So, there is any distinction between the privacy issues from this protocol compared to regular network operation and management, abuse reporting, etc. ?

TBD: In the case of content providers reporting, something like Network Error Logging and/or Navigation Timing could help for content providers in a way where more detailed information can be sent to an endpoint within a TLS connection in a way that isn't exposing anything to the network.?

7. Security Considerations

This document does not have any specific security considerations.

8. IANA Considerations

This document does not have any IANA considerations.

9. Acknowledgements

The author would like to acknowledge the inputs of Gert Doering, Erik Nygren ...

10. References

10.1. Normative References

[I-D.ietf-happy-happyeyeballs-v3]
Pauly, T., Schinazi, D., Jaju, N., and K. Ishibashi, "Happy Eyeballs Version 3: Better Connectivity Using Concurrency", Work in Progress, Internet-Draft, draft-ietf-happy-happyeyeballs-v3-02, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-happy-happyeyeballs-v3-02>.

10.2. Informative References

[I-D.trammell-happy-sad]
Trammell, B., "Slow Alternate Detection for Happy Eyeballs", Work in Progress, Internet-Draft, draft-trammell-happy-sad-00, 7 November 2025, <https://datatracker.ietf.org/doc/html/draft-trammell-happy-sad-00>.

[NEL] "Network Error Logging", <https://w3c.github.io/network-error-logging/>.

[RFC5424] Gerhards, R., "The Syslog Protocol", RFC 5424, DOI 10.17487/RFC5424, March 2009, <https://www.rfc-editor.org/info/rfc5424>.

[RFC5426] Okmianski, A., "Transmission of Syslog Messages over UDP", RFC 5426, DOI 10.17487/RFC5426, March 2009, <https://www.rfc-editor.org/info/rfc5426>.

Authors' Addresses

Jordi Palet Martinez
The IPv6 Company
Molino de la Navata, 75
28420 La Navata - Galapagar Madrid
Spain
Email: jordi.palet@theipv6company.com
URI: http://www.theipv6company.com/

Philipp S. Tiesel
SAP SE
Email: philipp@tiesel.net